

# ASG-Insight™ Reference Guide

Version: 6.0

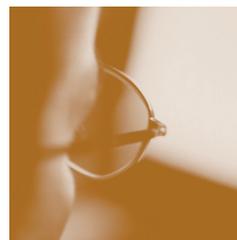
Publication Number: INX0400-60

Publication Date: February 2002

The information contained herein is the confidential and proprietary information of Allen Systems Group, Inc. Unauthorized use of this information and disclosure to third parties is expressly prohibited. This technical publication may not be reproduced in whole or in part, by any means, without the express written consent of Allen Systems Group, Inc.

© 1987-2002 Allen Systems Group, Inc. All rights reserved.

All names and products contained herein are the trademarks or registered trademarks of their respective holders.



ASG Worldwide Headquarters Naples, Florida USA | [asg.com](http://asg.com)

1333 Third Avenue South, Naples, Florida 34102 USA Tel: 941.435.2200 Fax: 941.263.3692 Toll Free: 1.800.932.5536







# ASG Support Numbers

ASG provides support throughout the world to resolve questions or problems regarding installation, operation, or use of our products. We provide all levels of support during normal business hours and emergency support during non-business hours. To expedite response time, please follow these procedures.

## Please have this information ready:

- Product name, version number, and release number
- List of any fixes currently applied
- Any alphanumeric error codes or messages written precisely or displayed
- A description of the specific steps that immediately preceded the problem
- The severity code (ASG Support uses an escalated severity system to prioritize service to our clients. The severity codes and their meanings are listed below.)
- Verify whether you received an ASG Service Pack for this product. It may include information to help you resolve questions regarding installation of this ASG product. The Service Pack instructions are in a text file on the distribution media included with the Service Pack.

## If You Receive a Voice Mail Message:

- 1 Follow the instructions to report a production-down or critical problem.
- 2 Leave a detailed message including your name and phone number. A Support representative will be paged and will return your call as soon as possible.
- 3 Please have the information described above ready for when you are contacted by the Support representative.

## Severity Codes and Expected Support Response Times

Severity	Meaning	Expected Support Response Time
1	Production down, critical situation	Within 30 minutes
2	Major component of product disabled	Within 2 hours
3	Problem with the product, but customer has work-around solution	Within 4 hours
4	"How-to" questions and enhancement requests	Within 4 hours

ASG provides software products that run in a number of third-party vendor environments. Support for all non-ASG products is the responsibility of the respective vendor. In the event a vendor discontinues support for a hardware and/or software product, ASG cannot be held responsible for problems arising from the use of that unsupported version.

## ***Business Hours Support***

<b>Your Location</b>	<b>Phone</b>	<b>Fax</b>	<b>E-mail</b>
<b>United States and Canada</b>	800.354.3578	941.263.2883	support@asg.com
<b>Australia</b>	61.2.9460.0411	61.2.9460.0280	support.au@asg.com
<b>England</b>	44.1727.736305	44.1727.812018	support.uk@asg.com
<b>France</b>	33.141.028590	33.141.028589	support.fr@asg.com
<b>Germany</b>	49.89.45716.222	49.89.45716.400	support.de@asg.com
<b>Singapore</b>	65.332.2922	65.337.7228	support.sg@asg.com
<b>All other countries:</b>	1.941.435.2200		support@asg.com

## ***Non-Business Hours - Emergency Support***

<b>Your Location</b>	<b>Phone</b>	<b>Your Location</b>	<b>Phone</b>
<b>United States and Canada</b>	800.354.3578		
<b>Asia</b>	65.332.2922	<b>Japan/Telecom</b>	0041.800.9932.5536
<b>Australia</b>	0011.800.9932.5536	<b>Netherlands</b>	00.800.3354.3578
<b>Denmark</b>	00.800.9932.5536	<b>New Zealand</b>	00.800.9932.5536
<b>France</b>	00.800.3354.3578	<b>Singapore</b>	001.800.3354.3578
<b>Germany</b>	00.800.3354.3578	<b>South Korea</b>	001.800.9932.5536
<b>Hong Kong</b>	001.800.9932.5536	<b>Sweden/Telia</b>	009.800.9932.5536
<b>Ireland</b>	00.800.9932.5536	<b>Switzerland</b>	00.800.9932.5536
<b>Israel/Bezeq</b>	014.800.9932.5536	<b>Thailand</b>	001.800.9932.5536
<b>Japan/IDC</b>	0061.800.9932.5536	<b>United Kingdom</b>	00.800.3354.3578
		<b>All other countries</b>	1.941.435.2200

## ASG Web Site

Visit <http://www.asg.com>, ASG's World Wide Web site.

Submit all product and documentation suggestions to ASG's product management team at <http://www.asg.com/asp/emailproductsuggestions.asp>.

If you do not have access to the web, FAX your suggestions to product management at (941) 263-3692. Please include your name, company, work phone, e-mail ID, and the name of the ASG product you are using. For documentation suggestions include the publication number located on the publication's front cover.



---

# Contents

---

<b>Preface</b> .....	<b>ix</b>
<b>About this Publication</b> .....	<b>ix</b>
<b>Related Publications</b> .....	<b>x</b>
<b>ASG-Existing Systems Workbench (ASG-ESW)</b> .....	<b>xi</b>
<b>Invoking ESW Products</b> .....	<b>xiv</b>
<b>ESW Product Integration</b> .....	<b>xv</b>
<b>Examples</b> .....	<b>xvi</b>
<b>Publication Conventions</b> .....	<b>xviii</b>
<b>1 Introduction</b> .....	<b>1</b>
<b>Insight Overview</b> .....	<b>1</b>
<b>Benefits</b> .....	<b>2</b>
<b>2 File</b> .....	<b>3</b>
<b>File Pull-down</b> .....	<b>4</b>
<b>File - Open Program Request Pop-Up</b> .....	<b>5</b>
<b>AKR Program Selection Pop-up</b> .....	<b>7</b>
<b>File - Close Program Request Pop-up</b> .....	<b>10</b>
<b>File - Save Program Request Pop-up</b> .....	<b>11</b>
<b>File - Analyze Submit Pop-up</b> .....	<b>12</b>
<b>File - AKR Utility Pop-up</b> .....	<b>15</b>
<b>File - AKR Directory Pop-up</b> .....	<b>17</b>
<b>File - AKR Allocate/Expand Pop-up</b> .....	<b>20</b>
<b>File - ASG-SmartEdit Request Pop-up</b> .....	<b>22</b>
<b>File - Execute Script Pop-up</b> .....	<b>23</b>

<b>3 View</b> .....	<b>25</b>
<b>View Pull-down</b> .....	<b>27</b>
<b>Source View Screen</b> .....	<b>29</b>
<b>View - Tree View Request Pop-up</b> .....	<b>30</b>
<b>Tree View Screen</b> .....	<b>33</b>
Tree View Generated Tags .....	33
Primary Commands.....	34
Line Commands .....	36
<b>View - Structure View Request Pop-up</b> .....	<b>37</b>
<b>Perform Structure View Screen</b> .....	<b>40</b>
<b>View - Paragraph Cross-Reference Request Pop-up</b> .....	<b>42</b>
<b>View - Paragraph Cross Reference Pop-up</b> .....	<b>45</b>
<b>View - Program Summary Pop-up</b> .....	<b>47</b>
<b>Program Summary - Files Pop-up</b> .....	<b>48</b>
<b>Program Summary - Data Pop-up</b> .....	<b>50</b>
<b>Program Summary - Copy Members Pop-up</b> .....	<b>51</b>
<b>Program Summary - Sections/Paragraphs Pop-up</b> .....	<b>52</b>
<b>Program Summary - Control Flow Pop-up</b> .....	<b>54</b>
<b>Program Summary - Preprocessor Statements Pop-up</b> .....	<b>55</b>
<b>View - Levels Request Pop-up</b> .....	<b>57</b>
<b>View - Reset Request Pop-up</b> .....	<b>58</b>
<b>View - Exclude Request Pop-up</b> .....	<b>59</b>
<b>4 Search</b> .....	<b>61</b>
<b>Search Actions</b> .....	<b>62</b>
<b>Search Pull-down</b> .....	<b>65</b>
<b>Search - Data Name Pop-up</b> .....	<b>66</b>
<b>Search - Label Name Pop-up</b> .....	<b>70</b>
<b>Search - Paragraph Name Pop-up</b> .....	<b>72</b>
<b>Search - Pattern String Pop-up</b> .....	<b>75</b>
<b>Search - COBOL Subset Name Pop-up</b> .....	<b>78</b>
<b>Search - Perform Range Name Pop-up</b> .....	<b>81</b>
<b>Search - Program Name Pop-up</b> .....	<b>84</b>

Search - User Mark Name Pop-up .....	86
Search - Line Range Pop-up.....	88
Search - Any/Unknown Type Pop-up .....	91
Search - Branch Request Pop-up.....	93
Branch Previous Options Pop-up .....	95
Selection List Pop-ups.....	96
IN-Clause Option Pop-up.....	97
<b>5 Logic .....</b>	<b>99</b>
Logic Pull-down.....	101
Logic Order Search - Data Name Pop-up.....	102
Logic Order Search - Label Name Pop-up .....	105
Logic Order Search - COBOL Subset Name Pop-up .....	107
Logic Order Search - Perform Range Name Pop-up.....	110
Logic Order Search - User Mark Name Pop-up .....	112
Logic Order Search - Subprogram Name Pop-up.....	114
Logic Order Search - Line Number Range Pop-up.....	116
Selection List Pop-ups.....	118
Logic Order Search - Trace IN-Clause Option Pop-up.....	119
<b>6 Task .....</b>	<b>121</b>
Task Pull-down .....	122
Automated Task - Modify Program Output Pop-up.....	123
Modify Output Task - Output Selection Pop-up .....	124
Modify Output Task - Statement Selection Pop-up.....	126
Modify Output Task - Data Name Selection Pop-up.....	127
Automated Task - Modify a Computation Pop-up .....	129
Modify Computation Task - Data Name Entry Pop-up .....	130
Modify Computation Task - Data Name Selection Pop-up .....	131
Modify Computation Task - Statement Selection Pop-up .....	132
Automated Task - Debug a Program Abend pop-up.....	134
Debug Abend Task - Line Number Entry Pop-up .....	134
Debug Abend Task - Statement Selection Pop-up .....	135

Automated Task - Result Options Pop-up .....	137
Options - COPY/Include Libraries Pop-up .....	138
<b>7 List</b> .....	<b>139</b>
List Pull-down .....	140
List - CALL Statements Pop-up .....	141
List - Equates Pop-up .....	142
List - User Marks Pop-up .....	143
List - Perform Range Names Pop-up .....	145
List - Program/Subprogram Names Pop-up .....	146
List - COBOL Subset Names Pop-up .....	147
<b>8 Options</b> .....	<b>149</b>
Options Pull-down .....	149
Options - Product Parameters Pop-up .....	151
Options - Product Allocations Pop-up .....	152
Options - Log/List/Punch Definition Pop-up .....	154
Options - Script File Allocations Pop-up .....	157
Options - PF Key Definition Pop-up .....	158
Options - Processing Modes Pop-up .....	160
Options - Scratchpad Pop-up .....	161
Options - Scratchpad Equate Pop-up .....	162
Options - Scratchpad Mark Pop-up .....	164
Options - Scratchpad Copy Pop-up .....	165
Options - Scratchpad Delete Pop-up .....	167
Options - Scratchpad Merge Pop-up .....	168
Options - Scratchpad Rename Pop-up .....	169
<b>9 Help</b> .....	<b>171</b>
Help Pull-down .....	172
Help - Specific ASG Command Pop-up .....	174
Help - Specific ASG Message Number Pop-up .....	175
Help - About Pop-up .....	175

<b>10 Commands</b> .....	<b>177</b>
<b>Command Processing</b> .....	<b>180</b>
<b>Recalling/Repeating Commands</b> .....	<b>181</b>
<b>Cursor Position</b> .....	<b>182</b>
<b>Cursor Substitution Character</b> .....	<b>183</b>
<b>Command Diagrams</b> .....	<b>183</b>
<b>&amp; (Retain) Command</b> .....	<b>186</b>
<b>ALLOCDEF Command</b> .....	<b>187</b>
<b>ANALYZE Command</b> .....	<b>188</b>
<b>BRANCH Command</b> .....	<b>189</b>
<b>CALL Command</b> .....	<b>192</b>
<b>CANCEL Command</b> .....	<b>193</b>
<b>COPY Command</b> .....	<b>194</b>
<b>CURRENT Command</b> .....	<b>197</b>
<b>DELETE Command</b> .....	<b>198</b>
<b>END Command</b> .....	<b>200</b>
<b>EQUATE Command</b> .....	<b>201</b>
<b>EXCLUDE Command</b> .....	<b>203</b>
<b>EXECUTE Command</b> .....	<b>209</b>
<b>FIND Command</b> .....	<b>212</b>
<b>FINDXTND Command</b> .....	<b>216</b>
<b>FLOW Command</b> .....	<b>228</b>
<b>GOBACK Command</b> .....	<b>233</b>
<b>HELP Command</b> .....	<b>234</b>
<b>HIGH Command</b> .....	<b>235</b>
<b>JUMP Command</b> .....	<b>242</b>
<b>KEYS Command</b> .....	<b>243</b>
<b>LEVELS Command</b> .....	<b>245</b>
<b>LIST Command</b> .....	<b>246</b>
List Menu Pop-up .....	<b>247</b>
<b>LOCATE Command</b> .....	<b>248</b>
<b>LPRINT Command</b> .....	<b>250</b>

<b>LPUNCH Command</b> .....	257
<b>MARK Command</b> .....	264
<b>MERGE Command</b> .....	267
<b>PARMDEF Command</b> .....	270
<b>PREF Command</b> .....	271
<b>PRINTLOG Command</b> .....	274
<b>PRINTLST Command</b> .....	275
<b>PROCESS Command</b> .....	276
<b>PRODLVL Command</b> .....	280
<b>PROGRAM Command</b> .....	281
<b>QUALIFY Command</b> .....	282
<b>RECALL Command</b> .....	283
<b>REDO Command</b> .....	286
<b>REFRESH Command</b> .....	288
<b>RENAME Command</b> .....	289
<b>REPEAT Command</b> .....	290
<b>RESET Command</b> .....	291
<b>RETURN Command</b> .....	292
<b>RFIND Command</b> .....	293
<b>RHIGH Command</b> .....	294
<b>RPREF Command</b> .....	295
<b>RSCROLL Command</b> .....	296
<b>RSTRUCTURE-VIEW Command</b> .....	297
<b>RTRACE Command</b> .....	298
Trace Decision Options Pop-up .....	299
<b>RTREEVW Command</b> .....	301
<b>SAVE Command</b> .....	302
<b>SCROLL Command</b> .....	303
<b>SELECT Command</b> .....	310
<b>SET Command</b> .....	311
LEARN Mode - Generated Command Pop-up .....	312
<b>SOURCEVIEW Command</b> .....	312

<b>STRUCTURE-VIEW Command</b> .....	<b>313</b>
<b>TRACE Command</b> .....	<b>315</b>
<b>TREEVIEW Command</b> .....	<b>321</b>
<b>VIEW Command</b> .....	<b>323</b>
<b>ZOOMIN/ZOOMOUT Commands</b> .....	<b>324</b>
<b>Line Commands</b> .....	<b>325</b>
F (First) Line Command .....	327
H (Highlight) Line Command .....	328
HH (Highlight Block) Line Command .....	329
J (Jump) Line Command.....	330
Label Line Command .....	330
L (Last) Line Command .....	331
S (Show) Line Command .....	332
SS (Show Block) Line Command.....	333
X (Exclude) Line Command.....	334
XX (Exclude Block) Line Command .....	335
ZI (Zoom In) Line Command .....	336
ZO (Zoom Out) Line Command .....	337
<b>11 Analyze</b> .....	<b>339</b>
<b>Overview</b> .....	<b>339</b>
<b>Analyzing a COBOL Program</b> .....	<b>340</b>
Analyze Input Descriptions.....	340
<b>The Analyze Process</b> .....	<b>342</b>
Analyze Using ISPF .....	343
Analyze Using ISPF/PDF Edit .....	343
Analyze Submit Parameters Screen .....	346
Options .....	346
<b>Automatic JCL Modifications</b> .....	<b>349</b>
<b>Analyze Summary Report</b> .....	<b>352</b>
<b>Analyze Options</b> .....	<b>354</b>
<b>12 AKR Utilities</b> .....	<b>361</b>
<b>Introduction to AKR Management</b> .....	<b>361</b>
<b>AKR Structure</b> .....	<b>362</b>
<b>Online AKR Utilities</b> .....	<b>362</b>

<b>Batch AKR Utilities</b> .....	<b>363</b>
Job Control Statements .....	364
Control Cards .....	364
<b>Command Format</b> .....	<b>365</b>
<b>Command Syntax</b> .....	<b>365</b>
Batch AKR Comments .....	367
CONVERT Batch AKR Command .....	367
COPY Batch AKR Command .....	368
DELETE Batch AKR Command .....	369
EXPORT Batch AKR Command .....	370
HELP Batch AKR Command .....	370
INIT Batch AKR Command .....	371
MOVE Batch AKR Command .....	371
PRINT Batch AKR Command .....	372
PUNCH Batch AKR Command .....	373
<b>Batch AKR Reports</b> .....	<b>375</b>
Allocating and Expanding AKRs without ISPF .....	377
<b>13 Paragraph Control Transfer Methods</b> .....	<b>383</b>
<b>14 Help Facility</b> .....	<b>391</b>
<b>Help Navigational Commands</b> .....	<b>393</b>
<b>Screen and Pop-up Help</b> .....	<b>394</b>
<b>Command Help</b> .....	<b>395</b>
<b>General Information</b> .....	<b>397</b>
<b>Specific Information</b> .....	<b>397</b>
<b>Help Abends</b> .....	<b>398</b>
<b>Help Messages</b> .....	<b>399</b>
Printing Messages .....	400
<b>Glossary</b> .....	<b>403</b>
<b>Index</b> .....	<b>409</b>

---

## Preface

---

This *ASG-Insight Reference Guide* provides instruction for using ASG-Insight (herein called Insight). Insight is a powerful interactive system automating the analysis process of COBOL programs.

Allen Systems Group, Inc. (ASG) provides professional support to resolve any questions or concerns regarding the installation or use of any ASG product. Telephone technical support is available around the world, 24 hours a day, 7 days a week.

ASG welcomes your comments, as a preferred or prospective customer, on this publication or on any ASG product.

## About this Publication

This publication consists of these chapters:

- [Chapter 1, "Introduction,"](#) provides an overview of Insight.
- [Chapter 2, "File,"](#) describes commands available on Insight's File menu.
- [Chapter 3, "View,"](#) describes commands available on Insight's View menu.
- [Chapter 4, "Search,"](#) describes commands available on Insight's Search menu.
- [Chapter 5, "Logic,"](#) describes commands available on Insight's Logic menu.
- [Chapter 6, "Task,"](#) describes commands available on Insight's Task menu.
- [Chapter 7, "List,"](#) describes commands available on Insight's List menu.
- [Chapter 8, "Options,"](#) describes commands available on Insight's Options menu.
- [Chapter 9, "Help,"](#) describes commands available on Insight's Help menu.
- [Chapter 10, "Commands,"](#) describes commands available in Insight.
- [Chapter 11, "Analyze,"](#) describes the analyze process.

- [Chapter 12, "AKR Utilities,"](#) describes how to use AKR utilities.
- [Chapter 13, "Paragraph Control Transfer Methods,"](#) describes how control is passed to and from target paragraphs.
- [Chapter 14, "Help Facility,"](#) describes how to use the comprehensive online help facility.

## Related Publications

The documentation library for ASG-Insight consists of these publications (where *nn* represents the product version number):

- *ASG-Center Installation Guide* (CNX0300-*nn*) contains installation and maintenance information for ASG-Center, the common set of libraries shared by the ASG-Existing Systems Workbench (ESW) suite of products.
- *ASG-Insight Installation Guide* (INX0300-*nn*) provides instruction about the installation and maintenance of ASG-Insight.
- *ASG-Insight Quick Reference Card* (INX0900-*nn*) provides a summary of the commands and functions within ASG-Insight.
- *ASG-Insight Quick Reference Guide* (INX0600-*nn*) summarizes the syntax and use of ASG-Insight commands.
- *ASG-Insight Reference Guide* (INX0400-*nn*) provides detailed information about the pop-ups, screens, and commands used in ASG-Insight, including field descriptions, usage notes, and command syntax.
- *ASG-Insight User's Guide* (INX0200-*nn*) provides user information for ASG-Insight.

**Note:** \_\_\_\_\_

To obtain a specific version of a publication, contact the ASG Service Desk.  
\_\_\_\_\_



This table contains the name and description of each ESW component:

ESW Product	Herein Called	Description
ASG-Alliance	Alliance	The application understanding component that is used by IT professionals to conduct an analysis of every application in their environment. Alliance supports the analysis and assessment of the impact of change requests upon an entire application. Alliance allows the programmer/analyst to accurately perform application analysis tasks in a fraction of the time it would take to perform these tasks without an automated analysis tool. The impact analysis from Alliance provides application management with additional information for use in determining the resources required for application changes.
ASG-AutoChange	AutoChange	The COBOL code change tool that makes conversion teams more productive by enabling quick and safe changes to be made to large quantities of code. AutoChange is an interactive tool that guides the user through the process of making source code changes.
ASG-Bridge	Bridge	The bridging product that enables field expansion for program source code, without being required to simultaneously expand the fields in files or databases. Because programs are converted in smaller groups, or on a one-by-one basis, and do not require file conversion, testing during the conversion process is simpler and more thorough.
ASG-Center	Center	The common platform for all ESW products. Center provides the common Analytical Engine to analyze the source program and store this information in the AKR. This common platform provides a homogeneous environment for all ESW products to work synergistically.

ESW Product	Herein Called	Description
ASG-Encore	Encore	The program re-engineering component for COBOL programs. Encore includes analysis facilities and allows you to extract code based on the most frequently used re-engineering criteria. The code generation facilities allow you to use the results of the extract to generate a standalone program, a callable module, a complement module, and a CICS server. Prior to code generation, you can view and modify the extracted Logic Segment using the COBOL editor.
ASG-Estimate	Estimate	The resource estimation tool that enables the user to define the scope, determine the impact, and estimate the cost of code conversion for COBOL, Assembler, and PL/I programs. Estimate locates selected data items across an application and determines how they are used (moves, arithmetic operations, and compares). Time and cost factors are applied to these counts, generating cost and personnel resource estimates.
ASG-Insight	Insight	The program understanding component for COBOL programs. Insight allows programmers to expose program structure, identify data flow, find program anomalies, and trace logic paths. It also has automated procedures to assist in debugging program abends, changing a computation, and resolving incorrect program output values.
ASG-Recap	Recap	The portfolio analysis component that evaluates COBOL applications. Recap reports provide function point analysis and metrics information, program quality assessments, intra-application and inter-application comparisons and summaries, and historical reporting of function point and metrics information. The portfolio analysis information can also be viewed interactively or exported to a database, spreadsheet, or graphics package.
ASG-SmartDoc	SmartDoc	The program documentation component for COBOL programs. SmartDoc reports contain control and data flow information, an annotated source listing, structure charts, program summary reports, exception reports for program anomalies, and software metrics.

ESW Product	Herein Called	Description
ASG-SmartEdit	SmartEdit	The COBOL editing component that can be activated automatically when the ISPF/PDF Editor is invoked. SmartEdit provides comprehensive searching, inline copybook display, and syntax checking. SmartEdit allows you to include an additional preprocessor (for example, the APS generator) during syntax checking. SmartEdit supports all versions of IBM COBOL, CICS, SQL, and CA-IDMS.
ASG-SmartTest	SmartTest	The testing/debugging component for COBOL, PL/I, Assembler, and APS programs in the TSO, MVS Batch, CICS (including file services), and IMS environments. SmartTest features include program analysis commands, execution control, intelligent breakpoints, test coverage, pseudo code with COBOL source update, batch connect, disassembled object code support, and full screen memory display.

## Invoking ESW Products

The method you use to invoke an ESW product depends on your system setup. If you need assistance to activate a product, see your systems administrator. If your site starts a product directly, use the ISPF selection or CLIST as indicated by your systems administrator. If your site uses the ESW screen to start a product, initiate the ESW screen using the ISPF selection or CLIST as indicated by your systems administrator and then typing in the product command on the command line.

The product names can also vary depending on whether you access a product directly or through ESW. See ["ESW Product Integration" on page xv](#) for more information about using ESW.

To initialize ESW products from the main ESW screen, select the appropriate option on the action bar pull-downs or type the product shortcut on the command line.

Product Name	Shortcut	ESW Pull-down Options
Alliance	AL	Understand ▶ Application
AutoChange	CC	Change ▶ Conversion Set
Bridge	BR	Change ▶ ASG-Bridge
Encore (Re-engineer)	EN	Re-engineer ▶ Program
Estimate	ES	Measure ▶ ASG-Estimate
Insight (Understand)	IN	Understand ▶ Program
Recap (Portfolio Analysis)	RC	Measure ▶ Portfolio
SmartDoc (Document)	DC	Document ▶ Program
SmartEdit	SE	Change ▶ Program <b>Or</b> Change ▶ Program with Options
SmartTest	ST	Test ▶ Module/Transaction

## ESW Product Integration

Because ESW is an integrated suite of products, you are able to access individual ESW products directly or through the main ESW screen. As a result, you might see different fields, values, action bar options, and pull-down options on a screen or pop-up depending on how you accessed the screen or pop-up.

Certain ESW products also contain functionality that interfaces with other ESW products. Using SmartTest as an example, if Alliance is installed, SmartTest provides a dynamic link to Alliance that can be used to display program analysis information. If Insight is installed and specified during the analyze, the Insight program analysis functions are automatically available for viewing logic/data relationships and execution path. For example, the Scratchpad option is available on the Options pull-down if you have Insight installed. Access to these integrated products requires only that they be installed and executed in the same libraries.



**Example 2.** [Figure 4](#) shows the File - Analyze Submit pop-up that displays when you access SmartTest directly. [Figure 5](#) shows the File - Analyze Submit pop-up that displays when you access SmartTest through ESW.

Notice that the Analyze features field in [Figure 5](#) lists additional ESW products than shown on [Figure 4](#). This field is automatically customized to contain the ESW products you have installed on your system.

The actions shown on these screens also vary. For example, the D action (ASG-SmartDoc Options) is available on the File - Analyze Submit screen if the SmartDoc product is installed on your system. In [Figure 4](#), the ASG-SmartDoc Options action is not available.

**Figure 4 • File - Analyze Submit Screen**

```

                                File - Analyze Submit
Command ==> -----
                E - Edit JCL                      S - Submit JCL

Compile and link JCL (PDS or sequential):
  Data set name 'USER12.REL.CNTL(UIAPCOBC)'

Analyze features (Y/N):
  ASG-SmartTest: Y   Extended Analysis: N

AKR data set name 'USER12.GENERAL.AKR'
AKR program name      (if overriding PROGRAM-ID)

Analyze options:
-----
-----

Compile? (Y/N) . . . . . Y   (Y if needed by features)
Link load Module reusable? (Y/N) Y
  
```

**Figure 5 • File - Analyze Submit Screen (Accessed through ESW)**

```

                                File - Analyze Submit
Command ==> -----
                E - Edit JCL   S - Submit JCL   D - ASG-SmartDoc Options

Compile and link JCL (PDS or sequential):
  Data set name 'USER12.REL.CNTL(HTEST)'

Analyze features (Y/N):
  ASG-Insight: Y   ASG-SmartTest: Y   Extended Analysis: N
  ASG-SmartDoc: N   ASG-Encore: N

AKR data set name 'USER12.GENERAL.AKR'
AKR program name      (if overriding PROGRAM-ID)

Analyze options:
-----
-----

Compile? (Y/N) . . . . . Y   (Y if needed by features)
Link load Module reusable? (Y/N) Y   (ASG-SmartTest)
  
```

## Publication Conventions

ASG uses these conventions in technical publications:

Convention	Represents
ALL CAPITALS	Directory, path, file, dataset, member, database, program, command, and parameter names.
Initial Capitals on Each Word	Window, field, field group, check box, button, screen, option names, and names of keys. A plus sign (+) is inserted for key combinations (e.g., Alt+Tab).
<i>lowercase italic monospace</i>	Information that you provide according to your particular situation. For example, you would replace <i>filename</i> with the actual name of the file.
Monospace	Characters you must type exactly as they are shown. Code, JCL, file listings, or command/statement syntax. Also used for denoting brief examples in a paragraph.
Vertical Separator Bar ( ) with underline	Options available with the default value underlined (e.g., Y  <u>N</u> ).

---

# 1

## Introduction

---

Insight is the ESW component that is designed to help you understand COBOL programs. Insight exposes program structure, identifies data flow, finds program anomalies, and traces logic paths. It also has automated procedures for debugging program abends, changing computations, and resolving incorrect program output values.

### Insight Overview

Insight analyzes COBOL programs using IBM's TSO/ISPF interface, and is an electronic window that lets you see the internal workings of a COBOL program.

Insight features Common User Access (CUA) screens, pull-downs, and pop-ups designed to give you easy access to all product features.

An action bar is the line of keywords displayed at the top of a screen. Each keyword represents an action category performed on that screen. To view the action list on a pull-down, move the cursor to the desired keyword and press Enter. To make a selection, type the number that corresponds to the desired action and press Enter. The result is either a generated command or a pop-up.

A pop-up is a window that displays on your screen that allows you to enter information for the requested action.

**Note:** \_\_\_\_\_  
See the *ASG-Insight User's Guide* for more information.

Powerful Insight facilities provide a precise, consistent, and thorough analysis of COBOL source code. You no longer need to summarize information from source listings, cross-references, and primitive online browse facilities. Each task is automated, yet blends with your current maintenance practices. You can determine what information you want and when you want it. You can also direct whether an overview or the details of a program display.

## Benefits

Insight analyzes COBOL source code rapidly and accurately.

Use Insight's flexible options to get an overview of both structured and unstructured programs. Insight gives you online access to complete data cross-referencing. These references include automatic checks for group level and redefined datanames. In addition, Insight retains the results so you can further evaluate them with other commands.

These Insight features reduce effort, eliminate oversights, and save time:

**Efficiency.** Collects information automatically from limited listings and browse facilities. Use an Insight action bar selection or a command to evaluate your program and to receive concise results.

**Accuracy.** Evaluates COBOL source code consistently and precisely.

**Usability.** Features easy-to-use CUA screens, pull-downs, and pop-ups eliminating the need to search and summarize program listings. Results are summarized in screen formats that prepare you for your next query.

**Accessibility.** Integrates accessibility into your daily TSO/ISPF work environment and supplies useful help facilities.

**Focus.** Manages technical details so that you can focus on understanding programs and assessing your changes.

**Understandability.** Provides pull-downs, pop-ups, and commands to automate each task needed to learn a program. Pop-ups or Insight commands easily reveal any COBOL program characteristic, and the resulting analysis is presented clearly.

Insight offers these five features that help you through the understanding process:

- Determining program structure
- Identifying data usage and relationships
- Tracing data usage in logic
- Tracing logic flow
- Saving, reusing, and printing information

---

# 2

## File

---

This chapter describes items on the Insight File menu and contains these sections:

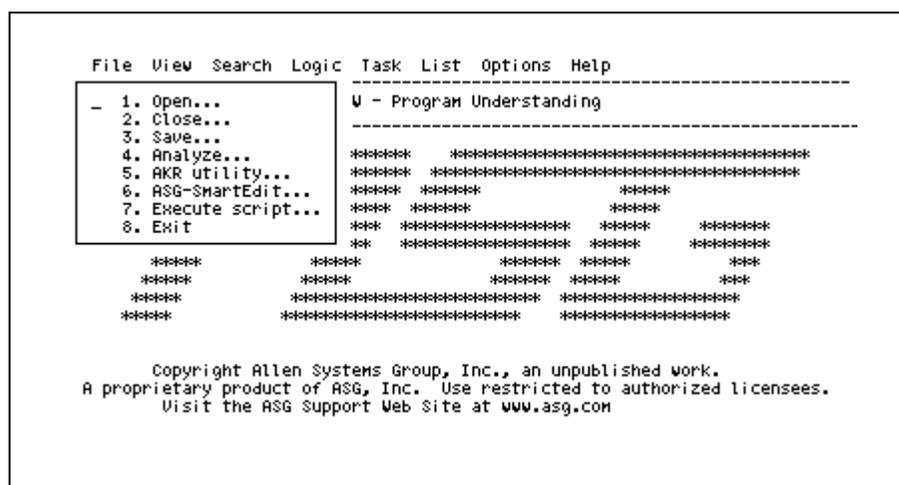
Topic	Page
<a href="#">File Pull-down</a>	<a href="#">4</a>
<a href="#">File - Open Program Request Pop-Up</a>	<a href="#">5</a>
<a href="#">AKR Program Selection Pop-up</a>	<a href="#">7</a>
<a href="#">File - Close Program Request Pop-up</a>	<a href="#">10</a>
<a href="#">File - Save Program Request Pop-up</a>	<a href="#">11</a>
<a href="#">File - Analyze Submit Pop-up</a>	<a href="#">12</a>
<a href="#">File - AKR Utility Pop-up</a>	<a href="#">15</a>
<a href="#">File - AKR Directory Pop-up</a>	<a href="#">17</a>
<a href="#">File - AKR Allocate/Expand Pop-up</a>	<a href="#">20</a>
<a href="#">File - ASG-SmartEdit Request Pop-up</a>	<a href="#">22</a>
<a href="#">File - Execute Script Pop-up</a>	<a href="#">23</a>

## File Pull-down

To access, edit, and analyze programs, manage the AKR, execute script files, and exit Insight, follow this step:

- ▶ Select File on the action bar and press Enter. The File pull-down, shown in [Figure 6](#), displays.

Figure 6 • File Pull-down



## Actions

Action	Result
1. Open	Displays the File - Open Program pop-up used to type the AKR and program name to be opened.
2. Close	Displays the File - Close pop-up used to exit the active program without ending the Insight session.
3. Save	Displays the File - Save pop-up used to save marks and/or equates in the active program.
4. Analyze	Displays the File - Analyze Submit pop-up used to submit an Analyze job, and to specify the AKR to store the analyze information.
5. AKR Utility	Displays the File - AKR Utility pop-up used to display the program directory, allocate or expand an AKR, rename a program, and delete a program.

Action	Result
6. ASG-SmartEdit	Displays the File - SmartEdit Request pop-up used to specify whether you want to edit or browse the source code.  This action only displays on the pull-down if SmartEdit is installed.
7. Execute script	Displays the File - Execute Script pop-up used to specify the dataset name and script option.
8. Exit	Closes the active program, ends the Insight session, and returns to the screen that you used to access Insight.  Use the values specified on the Options - Product Parameters pop-up to determine if equates and marks are to be saved when you select this action.

## File - Open Program Request Pop-Up

To open a program for examination in Insight, follow this step:

- ▶ Select File ▶ Open and press Enter. The File - Open Program Request pop-up, shown in [Figure 7](#), displays.

**Figure 7 • File - Open Program Request Pop-up**

```

File - Open Program Request

Type AKR information and program name. Then press Enter.

Application Knowledge Repository (AKR):

  Data set name . . 'VIAUSR.BRIDGE60.AKR'
  Program name . . _____ (blank for selection list)

  Volume serial . . _____ (if not cataloged)
  Password . . . . . (if password protected)

```

A program initially displays in Source View. Use the View pull-down to select other views. See ["Source View Screen" on page 29](#) and ["View" on page 25](#) for more information regarding views.

## Fields

Field	Action
Data set name	Required. Type the dataset name of the AKR where the program resides.
Program name	Optional. Type the name of the program if not entered in the Data set name field. If the program name is not entered, the AKR Program Selection pop-up displays to select the program name.
Volume serial	Required if the dataset is not cataloged. Type the volume serial number of the device where the AKR resides.
Password	Required if the dataset is protected. Type the dataset password.

**Note:**

After you enter the AKR information on the File - Open Program Request pop-up, it is saved and displays on the pop-up the next time it is opened.

## Usage Notes

When you open a new program while another program is open, the File - Open Program Request pop-up ([Figure 8 on page 6](#)) displays specifying the active program disposition and any updates. After you select an option, the AKR Program Selection pop-up displays allowing you to open a new program.

**Figure 8 • File - Open Program Request Pop-up When Active Program Exists**

```
File - Open Program Request

A program is currently open. Select an action to
be performed on the open program before opening
new program. Then press Enter.

1  1. Close the program, save any updates
   2. Close the program, no updates saved
   3. Do not close program
```

**Note:**

You can also open new programs with the CALL command, the PROGRAM command, or the VIEW command. See "[Commands](#)" on page 177 for more information on these commands.

## AKR Program Selection Pop-up

*To list information about each program in the AKR and choose a program*

- 1 Leave the program name on the File - Open Program pop-up blank and press Enter. The AKR Program Selection pop-up, shown in [Figure 9](#), displays.

**Figure 9 • AKR Program Selection Pop-up**

AKR Program Selection							
Command ==>						Scroll ==> CSR	
Application Knowledge Repository: USER.TEST.AKR							
S	Name	Alias Of	Type	Lines	Date	Time	Jobname
-	CALLDYN		IN	116	DDMMYYYY	HH:MM:SS	SRENRAPA
-	CALLLAB		IN	115	DDMMYYYY	HH:MM:SS	SRENRAPA
-	CHKACCT2		IN	310	DDMMYYYY	HH:MM:SS	VIADDBD
-	CTXP01		IN	12167	DDMMYYYY	HH:MM:SS	SRENLESD
-	DB2SAMP		IN	556	DDMMYYYY	HH:MM:SS	SRENRAPD
-	DB2001		IN	185	DDMMYYYY	HH:MM:SS	SRENRAPD
-	DB2002		IN	267	DDMMYYYY	HH:MM:SS	SRENRAPD
-	DB2003		IN	200	DDMMYYYY	HH:MM:SS	SRENRAPD
-	DB2004		IN	99	DDMMYYYY	HH:MM:SS	SRENRAPD
-	DECLTEST		IN	44	DDMMYYYY	HH:MM:SS	SRENRAPA
-	DECLTV		IN	39	DDMMYYYY	HH:MM:SS	SRENRAPA
-	DLITCBL	VIAPGM3	IN	248	DDMMYYYY	HH:MM:SS	SRENRAPA
-	EMPINQ		IN	1151	DDMMYYYY	HH:MM:SS	SRENLESD
-	IDMST001		IN	596	DDMMYYYY	HH:MM:SS	SRENRAPI
-	IDMST002		IN	443	DDMMYYYY	HH:MM:SS	SRENRAPI

- 2 Select a program using the line command area or type SELECT in the command input area.

More than one Insight user can view a COBOL program at the same time. The first person who enters a program has write access and can save marks and equates. All subsequent users cannot save marks and equates and receive a message that the program is currently being viewed.

You can scroll the AKR Program Selection pop-up using either the scroll commands or the LOCATE command. The LOCATE command scrolls directly to the specified name. If the name does not exist, the scroll facility displays the nearest match.

## Fields

Field	Description
Application Knowledge Repository	Specifies the AKR dataset name specified on the File - Open Program Request pop-up.
Line command area	Identifies the line command area, which accepts these line commands:  S Selects the specified program. The Source View screen displays.  / Locates the specified program. The screen is scrolled until the specified program is at the top of the display.
Name	Specifies the name of the program in the AKR, from the PROGRAM-ID statement.  If the analyzed program contains an ENTRY point, Name is the ENTRY point name.  If you overrode the name in the PROGRAM-ID statement when you submitted the analyze job, Name is the name you entered in the AKR program name field on the File - Analyze Submit pop-up. See <a href="#">"File - Analyze Submit Pop-up" on page 12</a> or <a href="#">"File - AKR Directory Pop-up" on page 17</a> for more information about AKR entries.
Alias Of	Specifies the name of the program containing the ENTRY point, if the analyzed program contains an ENTRY point.  If you overrode the name in the PROGRAM-ID statement when you submitted the analyze job, Alias of is the name that you entered in the AKR program name field on the File - Analyze Submit pop-up. See <a href="#">"File - AKR Directory Pop-up" on page 17</a> for more information about alias names in the AKR.

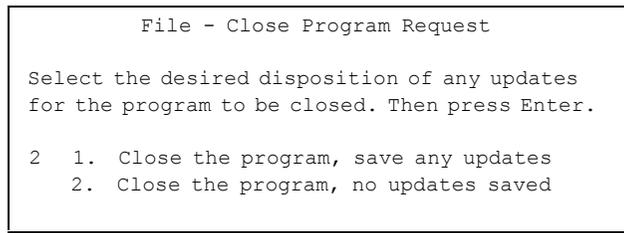
Field	Description
Type	<p>Indicates the analysis type(s) performed on the program. The File - Analyze Submit pop-up shows the analysis type when the analyze job is submitted. Insight only uses programs with the IN analysis type. These are the valid analysis types:</p> <p>EN Indicates an Encore analysis was performed.</p> <p>IN Indicates an Insight analysis was performed.</p> <p>ST Indicates a SmartTest analysis was performed.</p> <p>DS Indicates a SmartDoc analysis was performed.</p> <p>DC Indicates a SmartDoc analysis with a COBOL compile was performed.</p> <p>DX Indicates an Extended SmartDoc analysis was performed.</p> <p>DA Indicates an Extended SmartDoc analysis with a COBOL compile was performed.</p> <p>ASM Indicates the program is an Assembler source program.</p> <p>PROFILE Indicates that the member contains SmartTest profile information.</p> <p>METRICS Indicates that the member contains SmartDoc metric information.</p> <p>INTERNAL This is an Encore feature that indicates the member contains logic segment information.</p>
Lines	Lists the number of program source lines.
Date/Time	Lists the date and time the program was analyzed.
Jobname	Specifies the job name that analyzed the program.

## File - Close Program Request Pop-up

To exit the active program without exiting Insight and to save marks and equates, follow this step:

- ▶ Select File ▶ Close and press Enter. The File - Close Program Request pop-up, shown in [Figure 10](#), displays.

**Figure 10 • File - Close Program Request Pop-up**



Use the File - Save Program Request pop-up or the SAVE command to save marks and equates.

### Field

Field	Options
Option	<p>Overrides the values set on the Options - Product Parameters pop-up. If you select no option, Insight uses the values on the Options - Product Parameters pop-up to determine if marks and equates are to be saved. Choose one of these options:</p> <p>Close the program, save updates</p> <p>Exits the active program saving any marks and equates that you created or changed during the session. This overrides the values set on the Options - Product Parameters pop-up.</p> <p>Close the program, no updates saved</p> <p>Exits the active program without saving any marks and equates that you created or changed during the session. This overrides the values set on the Options - Product Parameters pop-up.</p>

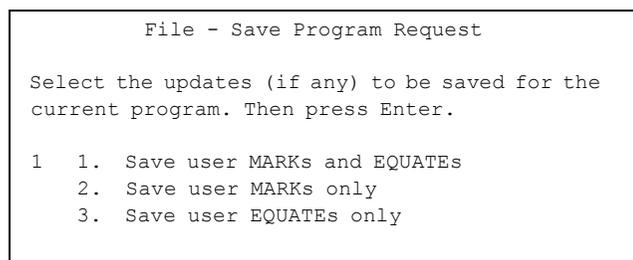
---

## File - Save Program Request Pop-up

To save the AKR marks and/or equates that were created or changed, without ending the session, follow this step:

- ▶ Select File ▶ Save and press Enter. The File - Save pop-up, shown in [Figure 11](#), displays.

**Figure 11 • File - Save Program Request Pop-up**



To save marks and equates, use the File - Close pop-up or the SAVE command.

### Field

Field	Description
Option	<p>Overrides the values set on the Options - Product Parameters pop-up. If you select no option, Insight uses the values set on the Options - Product Parameters pop-up to determine if marks and equates are to be saved.</p> <p>Choose one of these options:</p> <p>Save User MARKs and EQUATEs</p> <p>Select 1 to save all marks and equates for the active program. This overrides the values set on the Options - Product Parameters pop-up.</p> <p>Save User MARKs only</p> <p>Select 2 to save all marks created or changed for the active program. Equates are not saved. This overrides the values set on the Options - Product Parameters pop-up.</p> <p>Save User EQUATEs only</p> <p>Select 3 to save all equates entered for the active program. Marks are not saved. This overrides the values set on the Options - Product Parameters pop-up.</p>

### Usage Notes

See ["Options - Scratchpad Pop-up" on page 161](#) for information about creating user marks and equates.

## File - Analyze Submit Pop-up

To edit and submit JCL, to specify the AKR information for the analyze job, and to store the information on the AKR, follow this step:

- ▶ Select File ▶ Analyze, or type ANALYZE in any command input area, and press Enter. The File - Analyze Submit pop-up, shown in [Figure 12](#), displays.

**Note:**

You can also analyze programs using other methods. See "[Analyzing a COBOL Program](#)" on page 340 for more information.

**Figure 12 • File - Analyze Submit Pop-up**

```

ASG-ESW - Prepare Program
Command ===> _____
                E - Edit JCL      S - Submit JCL      D - Doc Options

Compile and link JCL (PDS or sequential):
  Data set name  'USER.TEST.CNTL(YOURJCL) '

Analyze features (Y/N):
  Understand: Y  Test: Y  Extended Analysis: Y  Document: N
  Re-engineer:  N
AKR data set name 'USER.TEST.AKR'
AKR program name _____ (if overriding PROGRAM-ID)

Analyze options:
  _____
  _____
  _____

Compile? (Y/N) . . . . . Y      (Y if needed by features)
Link load module reusable? (Y/N) Y      (Test only)
  
```

### Options

Option	Description
E - Edit JCL	<p>Enables you to review or change the compile/analyze JCL, if required. To do this, type E and press Enter. The JCL member you specified in the Data Set Name field generates the JCL to be edited, applying the rules outlined in the Analyze appendix. The Edit screen displays the generated JCL.</p> <p>Type SUBMIT to submit the edited JCL for execution when you have completed editing. Optionally, you can save the edited JCL in a partitioned dataset using the CREATE command. Otherwise, any changes made at this time are not saved.</p>

Option	Description
D - Doc Options	Displays only if SmartDoc is installed. Type D to display the File - SmartDoc Options screen used to request an Extended SmartDoc analysis and to specify what reports (if any) are generated.
S - Submit JCL	Submits the JCL to compile/analyze the specified program. The JCL submitted is generated from the JCL member specified in the Data Set name field, using the rules outlined in the Analyze appendix.

## Fields

Field	Description
Data set name	Required. The PDS member or sequential dataset containing the JCL to compile and link the program. If the JCL resides in a source manager such as Librarian or Panvalet, use the VIASUB edit macro to submit the compile/analyze job.
Understand	Displays only if Insight is installed. This type of analysis provides Insight logic and program execution flow capabilities. If Insight is the only product installed, this field contains YES and cannot be changed. The default is Y.
Test	Displays only if SmartTest is installed. Y indicates that a SmartTest compile/analysis will be performed. This type of analysis provides the testing and debugging information required by SmartTest. If SmartTest is the only product installed this field contains YES and cannot be changed. The default is Y.
Extended Analysis	Displays only if SmartTest is installed. This type of analysis provides comprehensive program analyzing capabilities in addition to the testing and debugging capabilities of SmartTest. The default is Y.  A SmartDoc extended analysis is specified on the File - SmartDoc Options screen.
Document	Displays only if SmartDoc is installed. This type of analysis provides the report information generated by SmartDoc. If SmartDoc is the only product installed this field contains YES and cannot be changed. The default is Y.
Re-engineer	Displays only if Encore is installed. This type of analysis provides the logic and program execution flow capabilities of Encore. YES specifies that an Encore compile/analysis is to be performed. The default is N.

Field	Description
AKR data set name	Required. Type the name of the AKR that is to contain the information for the analyzed program.
AKR program name	<p>Optional. Type an alias name to be used by the analyze process to save its results in the AKR.</p> <p>If you do not enter a value in this field, the analyze job saves the results in the AKR using the PROGRAM-ID statement name in the COBOL source.</p> <p>If you enter a AKR program name, the analyzed job saves the results using the entered name in the AKR, and also as an PROGRAM-ID alias.</p> <p>If a program contains ENTRY points, the analyze job also saves a member for each ENTRY point in the AKR, with an alias of the PROGRAM-ID.</p> <p><b>Note:</b> _____ This field is only used for the AKR program name and does not change the COBOL program name in the source.</p>
Analyze options	Optional. Type the analyze options you want to override. Default options for the analyze job are established at installation time. Analyze options that can be entered in this field are described in the Analyze appendix.
Compile?	Optional. You do not need to compile or link a program if Insight, Encore, or SmartDoc are the only analyze features specified. Type N in this field to suppress the JCL compile step and link steps. This field is forced to a value of Y if SmartTest and/or Extended analysis is selected.
Link load module reusable	Optional. Use this field to test a program under SmartTest that is dynamically loaded and is tested with RUN MONITOR. You must mark the load module as reusable to retain the Breakpoints across calls. The default is Y.

## File - AKR Utility Pop-up

To display the program directory, allocate or expand an AKR, rename a program, and delete a program, follow this step:

- ▶ Select File ▶ AKR Utility and press Enter. The File - AKR Utility pop-up, shown in [Figure 13](#), displays.

**Figure 13 • File - AKR Utility Pop-up**

```

ASG-ESW - AKR Utility
Command ==> _____

Blank - Display member list      D - Delete member
A   - Allocate/expand AKR      R - Rename member

Application Knowledge Repository (AKR):

Data set name . . 'VIAUSR.BRIDGE60.AKR'
Member . . . . . _____ (if "R" or "D" selected)
New name . . . . . _____ (if "R" selected)

Volume serial . . _____ (if not cataloged)
Password . . . . . _____ (if password protected)

```

### Options

Options	Description
Blank - Display member list	Displays the File - AKR Directory pop-up that lists the programs stored in the AKR. You can also delete or rename programs on the File - AKR Directory pop-up.
A - Allocate/expand AKR	Select option A to allocate a new AKR or to expand an existing AKR. Type the AKR name in the Data set name field to display the File - AKR Allocate/Expand pop-up. If expanding an existing AKR, type YES in the Expand existing AKR field on that pop-up.
D - Delete member	Select option D to delete a program. Prior to selecting option D, type the AKR name in the Data set name field and the program name in the Program field.
R - Rename member	Select option R to rename a program. Prior to selecting option R, type the AKR name in the Data set name field, the program name in the Program field, and the new name in the New name field.

## Fields

Field	Description
Data set name	Required. Type the name of the AKR directory. If the TSO ID qualifier is the same as the user ID, type the group, type, and program without quotes. If the TSO ID qualifier is different than the user ID, type the project, group, type, and program within quotes.
Member	Required when deleting or renaming a program; otherwise, you can enter the program name either in the Program field or in the Data set name field. Type the name of the program in the AKR.
New name	Required when renaming a program. Type the new name of the program. The name must be 1 to 10 alphanumeric characters.
Volume serial	Required if the dataset specified in the Data set name field is not cataloged. Type the volume serial number. If the dataset is cataloged, this field is optional.
Password	Required if the dataset is protected. Type the dataset password.

## File - AKR Directory Pop-up

To list all members in the specified AKR, follow this step:

- ▶ Leave the command input area blank on the File - AKR Utility pop-up and press Enter. The File - AKR Directory pop-up, shown in [Figure 14](#), displays.

**Figure 14 • File - AKR Directory Pop-up**

File - AKR Directory							
Command ==>				Scroll ==> CSR			
AKR: USER.TEST.AKR				Total members:	26	Total entries:	29
D - Delete R - Rename				Total records:	12000	Free space:	41.2%
Row	Name	New name	Alias of	Type	Date	Time	Jobname Space
-	\$\$SEGMENTS			INTERNAL	DDMMYYYY	HH:MM	SRENMDR 0.0%
-	AT810B0			IN,RN	DDMMYYYY	HH:MM	SRENDSMB 1.1%
-	DBMSCBL		ET070B0	IN,RN	DDMMYYYY	HH:MM	SRENDSMB
-	DBMSCBL		ET760B0	IN,RN	DDMMYYYY	HH:MM	SRENDSMB
-	DSM00022			IN,RN	DDMMYYYY	HH:MM	SRENDSMB 0.3%
-	ET070B0			IN,RN	DDMMYYYY	HH:MM	SRENDSMB 3.3%
-	ET760B0			IN,RN	DDMMYYYY	HH:MM	SRENDSMB 1.7%
-	ET927BJ			IN,RN	DDMMYYYY	HH:MM	SRENDSMB 2.8%
-	E02031			IN,RN	DDMMYYYY	HH:MM	SRENDSMB 1.6%
-	INDEXBUG			IN,RN	DDMMYYYY	HH:MM	SRENMDR2 0.3%
-	INDEXDEM			IN,RN	DDMMYYYY	HH:MM	SRENMDR9 0.3%
-	LSTRACE0			IN,RN	DDMMYYYY	HH:MM	SRENFKKB 0.3%
-	LSTRACE1			IN,RN	DDMMYYYY	HH:MM	SRENFKKB 0.3%

Scroll to view members that are not visible, or type L (Locate) plus a character string to scroll to a specific member. The pop-up scrolls to the member that most closely matches the character string.

### Fields

Field	Description
AKR	Specifies the complete dataset name of the requested AKR as entered on the File - AKR Utility pop-up.
Row	Specifies the relative number in the AKR of the first member displayed on this pop-up.
Total members	Specifies the total number of members in this AKR, not including aliases.
Total entries	Specifies the total number of members in this AKR, including aliases.
Total records	Specifies the number of records allocated to this AKR, as entered in the Space Amount field on the File - AKR Allocate/Expand pop-up when the AKR was either allocated or expanded.  If the Space Units was entered as Tracks or Cylinders on that pop-up instead of Records, the amount is converted to Records for display in the Total records field.

Field	Description
Free space	Indicates the amount of available space in this AKR, rounded to the nearest .1 percent.
Line command area	<p>Specifies the area to the left of the Name field, which accepts these commands:</p> <p>D - Delete Type <b>D</b> to the left of each program you want to delete from the AKR. You can also delete programs on the File - AKR Utility pop-up.</p> <p>R - Rename Type <b>R</b> to the left of the program you want to rename, and type the new name in the New name field. When you change the name of a primary program, the alias name is automatically changed. You can also rename programs on the File - AKR Utility pop-up.</p> <p><b>Note:</b> Alias programs cannot be deleted or renamed.</p>
Name	<p>Specifies the names of the member in the AKR. Program names are taken from the PROGRAM-ID statement. If the analyzed program contains an ENTRY point, Name is the ENTRY point name.</p> <p>If the name in the PROGRAM-ID statement was overridden at the time the analyze job was submitted, Name is the name that you entered in the AKR program name field on the File - Analyze Submit pop-up.</p> <p>See <a href="#">"File - AKR Directory Pop-up" on page 17</a> for more information about entries in the AKR.</p>
New name	Specifies the new member name. This field is required when you rename a member. The member name must be 1 to 10 alphanumeric characters.
Alias of	<p>Specifies the name of the program that contains the ENTRY point, if an analyzed program contains an ENTRY point.</p> <p>If the you override the name in the PROGRAM-ID statement at the time the analyze job was submitted, Alias of is the name that you entered in the AKR program name field on the File - Analyze Submit pop-up.</p> <p>An alias program cannot be deleted or renamed. The alias name is automatically changed when you either delete or rename the primary program.</p> <p>See <a href="#">"File - AKR Directory Pop-up" on page 17</a> for more information about alias names in the AKR.</p>

Field	Description
Type	<p>Indicates the type of analysis that was performed on the member and specified on the File - Analyze Submit pop-up. These are the valid types:</p> <p>AL All Alliance analysis was performed.</p> <p>ST A SmartTest analysis was performed.</p> <p>IN An Insight analysis was performed.</p> <p>DS A SmartDoc analysis was performed. DC - A SmartDoc analysis with a COBOL compile was performed.</p> <p>DX A SmartDoc Extended analysis was performed.</p> <p>DA A SmartDoc Extended analysis with a COBOL compile was performed.</p> <p>ASM The program is an Assembler source program.</p> <p>RC A Recap analysis was performed.</p> <p>EN An Encore analysis was performed.</p>
Type	<p>PROFILE This feature indicates the member contains profile information. Generally, the member name is the profile user ID. Profile information is automatically saved when you use the SmartTest Session Tailoring screen to specify the testing environment options. After you specify program level testing options on the Session Tailoring screen (and they are saved in the AKR profile member), the profile member is used by SmartTest when you initiate a test session. A profile is only used (and updated) by the user for whom it is created. The profile member automatically reflects any modifications to the Session Tailoring screen.</p> <p>METRICS This is a SmartDoc feature that indicates the member contains metrics information. The Name field contains \$\$METRIC or the name you assigned to the metrics using the Rename function. The \$\$METRIC member contains metrics calculated for the programs in this AKR.</p> <p>INTERNAL This is an Encore feature that indicates the member contains logic segment information. The Name field contains \$\$SEGMENTS or the name assigned to the member using the Rename function. The \$\$SEGMENTS member contains the logic segment information.</p>
Date	Specifies the date the program was analyzed.

Field	Description
Time	Specifies the time the program was analyzed.
Jobname	Specifies the Job name used to analyze the program.
Space	Specifies the percentage of space the program is using on this AKR. This percentage is rounded to the nearest .1 percent.

## File - AKR Allocate/Expand Pop-up

### To allocate a new AKR

- 1 From the ASG-ESW - AKR Utility pop-up, select option A and press Enter. The File - AKR Allocate/Expand pop-up, shown in [Figure 15](#), displays.

**Figure 15 • AKR Allocate/Expand Pop-up with SMS and a VSAM AKR**

```

Command ==> _____ File - AKR Allocate/Expand
_____
      S - Submit JCL      E - Edit JCL      C - Specify Catalog
Expand existing AKR . . . NO          (Yes or No)
AKR data set name . . . 'USER.TEST.AKR'
Management Class . . . _____ (Blank for default MGMTCLAS)
Storage Class . . . _____ (Blank for default STORCLAS)
Volume . . . _____
Data Class . . . _____ (Blank for default DATACLAS)
Space units . . . . . RECORDS (Records, Tracks or Cylinders)
Space amount . . . . . 4000 (Total AKR size in above units)
Unique . . . . . YES (Enter NO for a VSAM data space)

Job statement information:
//NAME      JOB (ACCOUNT),NAME,
//          MSGCLASS=A
//          INSERT '/*ROUTE PRINT NODE.USER' HERE IF NEEDED.
//          */
  
```

- 2 Type NO in the Expand existing AKR field and press Enter. The space the AKR needs depends on the size and the number of COBOL programs that are being analyzed and placed in it.

### To allocate a new AKR or to expand an existing AKR

- 1 From the ASG-ESW - AKR Utility pop-up, select option A and press Enter. The File - AKR Allocate/Expand pop-up, shown in [Figure 15](#), displays.
- 2 Type YES in the Expand existing AKR field and press Enter.

## Options

Option	Description
S - Submit JCL	Submits the generated JCL to allocate or expands the AKR shown in the AKR Data Set Name field.
E - Edit JCL	Enables you to edit the generated JCL used to allocate or to expand the AKR shown in the AKR data set name field.
C - Specify Catalog	Displays the AKR Catalog Information pop-up that is used to specify the dataset name and password of a catalog, if required.

## Fields

Field	Description
Expand existing AKR	Expands the AKR shown in the AKR data set name field. Type NO to allocate a new AKR. The default is NO.
AKR data set name	Specifies the AKR name specified on the File - AKR Utility pop-up.
Management Class	If your site uses SMS, type the management class for the AKR. The default is set during installation.
Storage Class	If your site uses SMS, type the storage class for the AKR. The default is set during installation.
Volume	Required if you are not using SMS. Type the volume serial number where the AKR is to reside.
Data Class	If your site uses SMS, type the data class for the AKR. The default is set during installation.
Space units	Optional. Do not enter if you are using SMS. Type the type of units to allocate. The units must be records, tracks, or cylinders. The default is records.
Space amount	Required if you are not using SMS. Type the amount of space to allocate in the type of units specified in the Space units field. The space needed for the AKR depends on the size and the number of programs being analyzed and placed in it. See the <i>ASG-Center Installation Guide</i> for space estimates.
Unique	Optional. Type NO if the AKR is a sub allocation in a VSAM data space on the volume. Otherwise, type YES. The default is YES.

Field	Description
Job statement information	Required to submit JCL. Type the appropriate JOB statement information for your site. If you need to specify either a private catalog or the password for your catalog, type C in the command input area to display the AKR Catalog Information pop-up. You can use this pop-up to specify the Catalog DSN and Password.
Catalog DSN	Optional. Type the catalog dataset name if you want the AKR dataset name to be added to a private catalog.
Password	Required if the CATALOG DSN is protected. Type the password for the catalog dataset name specified in the CATALOG DSN field.

## File - ASG-SmartEdit Request Pop-up

**Note:**

This pop-up is only available if SmartEdit is installed.

To browse or to edit the source code using SmartEdit, follow this step:

- ▶ Select File ▶ SmartEdit and press Enter. The File - ASG SmartEdit Request pop-up, shown in [Figure 16](#), displays.

**Figure 16 • File - ASG-SmartEdit Request Pop-up**

```
File - ASG-SmartEdit Request

Select Edit or Browse, and select the Options
Panel display option. Then press Enter.

Type of request      Options
1  1. Edit           /  Display options panel
   2. Browse
```

## Fields

Field	Description
Type of request	Either edits or browses the source code. The Edit or Browse Entry pop-up displays.
Display options panel	Displays the Edit or Browse Options pop-up before accessing the Edit or the Browse Entry pop-up. If not specified, the Edit or Browse Options pop-up is bypassed and the Edit or Browse Entry pop-up displays. The default is to display the Edit or Browse Options pop-up.

## File - Execute Script Pop-up

To begin processing a script file, follow this step:

- ▶ Select File ▶ Execute script and press Enter. The File - Execute Script pop-up, shown in [Figure 17](#), displays.

**Figure 17 • File - Execute Script Pop-up**

```

File - Execute Script

Type a data set name, a data set name with a member name, or just a member
name in parentheses. Select the "Single step" option if you wish to
execute the script one command at a time.

If you wish to "resume" normal execution of a "single step" script, select
the "Resume automatic mode" option. If you wish to cancel a "single step"
script, select the "Cancel execution" option. The "resume" and "cancel"
options should be entered with a blank data set name field.

Data set name _____

Options
1  1. Automatic
   2. Single step
   3. Resume automatic mode
   4. Cancel execution

```

## Fields

Field	Description
Data set name	Required. Specifies the name of the script file to be processed.
Options	<p>Specifies the processing mode for the script file. These are the three options:</p> <ul style="list-style-type: none"><li>• <b>Automatic</b> Each command in the script file is executed consecutively. This is the default.</li><li>• <b>Single Step</b> Each command in the script file displays in the command input area. You can either change or erase the displayed command. Press Enter to actually execute the displayed command. Processing of the script file continues in this manner until all commands have been displayed.</li><li>• <b>Resume Automatic Mode</b> After a script file has begun executing in Single step mode, selecting Resume executes the remaining script file commands without stepping through each.</li><li>• <b>Cancel execution</b> After a script file has begun executing in Single step mode, selecting Cancel stops the execution of the remaining script file commands.</li></ul> <p>Script files can also be executed using the EXECUTE command. See <a href="#">"EXECUTE Command" on page 209</a> for more information.</p>

---

---

# 3

## View

---

This chapter describes items on the Insight View menu and contains these sections:

Topic	Page
<a href="#">View Pull-down</a>	<a href="#">27</a>
<a href="#">Source View Screen</a>	<a href="#">29</a>
<a href="#">View - Tree View Request Pop-up</a>	<a href="#">30</a>
<a href="#">Tree View Screen</a>	<a href="#">33</a>
<a href="#">View - Structure View Request Pop-up</a>	<a href="#">37</a>
<a href="#">Perform Structure View Screen</a>	<a href="#">33</a>
<a href="#">View - Paragraph Cross-Reference Request Pop-up</a>	<a href="#">34</a>
<a href="#">View - Paragraph Cross Reference Pop-up</a>	<a href="#">36</a>
<a href="#">View - Program Summary Pop-up</a>	<a href="#">47</a>
<a href="#">Program Summary - Files Pop-up</a>	<a href="#">48</a>
<a href="#">Program Summary - Data Pop-up</a>	<a href="#">50</a>
<a href="#">Program Summary - Copy Members Pop-up</a>	<a href="#">51</a>
<a href="#">Program Summary - Sections/Paragraphs Pop-up</a>	<a href="#">52</a>
<a href="#">Program Summary - Control Flow Pop-up</a>	<a href="#">54</a>
<a href="#">Program Summary - Preprocessor Statements Pop-up</a>	<a href="#">55</a>
<a href="#">View - Levels Request Pop-up</a>	<a href="#">57</a>

Topic	Page
<a href="#">View - Reset Request Pop-up</a>	<a href="#">58</a>
<a href="#">View - Exclude Request Pop-up</a>	<a href="#">59</a>

Insight displays analyzed COBOL program information in source, graphical, and list formats. These formats give different views of the program and its logical components. Use these views to examine the program and to learn a programs structure and function.

The View pull-down has three primary view methods to assist you in isolating, understanding, and identifying the important sections of the source program. These are the types of views:

View	Description
Source view	Presents the actual COBOL source code in source statements order.
Tree view	Presents the paragraph hierarchy, that shows the program structure in logical execution order.
Structure view	Presents the program structure graphically.

You can change the magnification in all of the view methods (i.e., you can zoom in and out of logical program components). Each view is created from the same original COBOL source program, but presents a different perspective, and is concurrently available and synchronized with the other views. You may switch back and forth between views and retain the results of certain analysis inquiries.

The Structure and Tree Views display the logical program components and can be tailored to display varying program structure levels. Tailor these displays using the Request pop-ups for each.

You can request these program understanding list formats from any primary view:

Format	Description
The Paragraph Cross-reference	Shows the control flow between paragraphs in the program.
The Program Profile	Provides information about file, copy member, section, control, and database usage.

## View Pull-down

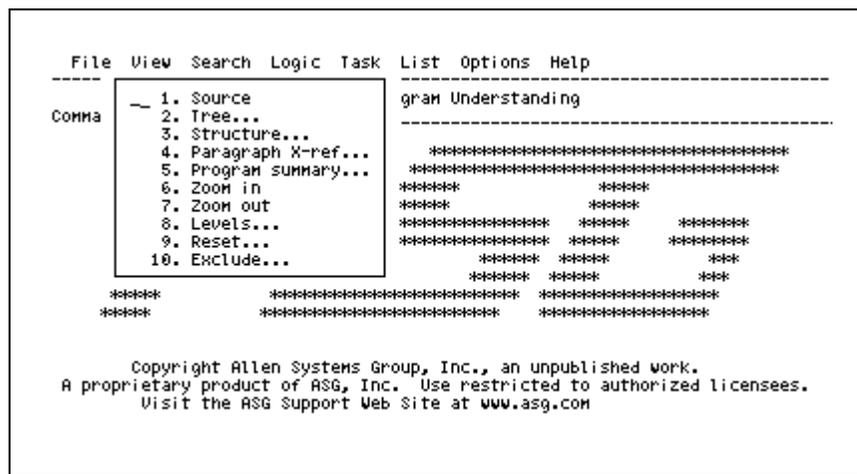
Use the View pull-down shown in [Figure 18](#) to perform these tasks:

- View a program
- View paragraph cross-reference information
- View program information
- Display or exclude source according to program hierarchical structure
- Change the information level viewed
- Reset the display
- Exclude lines from the display

To display the View pull-down, follow this step:

- ▶ Select View from the action bar and press Enter.

**Figure 18 • View Pull-down**



## Actions

Action	Description
1. Source	Displays the Source View screen, that shows the program in COBOL source statement order.
2. Tree	Displays the program in logical execution order, showing logical program units and the relationships between related units. When you choose this action, the View - Tree View Request pop-up displays allowing you to specify the format of the Tree View display.

Action	Description
3. Structure	Displays the program in a graphical format, with each box on the screen representing a logical program unit. When you choose this action, the View - Structure View Request pop-up displays allowing you to specify the format of the Structure View display.
4. Paragraph X-ref	Displays the View - Paragraph Cross Reference Request pop-up used to customize the View - Paragraph Cross Reference pop-up.
5. Program summary	Displays the View - Program Summary pop-up, used to select the type of summary information displayed.
6. Zoom in	Displays source lines according to the hierarchical levels of the program. When you select this action, a message displays to remind you to position the cursor and to press Enter.
7. Zoom out	Excludes source lines according to the hierarchical levels of the program. When you select this action, a message displays reminding you to position the cursor and press Enter. This action is not valid on the Perform Structure View screen.
8. Levels	Displays the View - Levels Request pop-up used to specify the new level number. This action is valid only on the Tree View and Perform Structure View screens.
9. Reset	Displays the View - Reset Request pop-up used to specify what display features to reset, including highlighting, tags, excluded lines, pending line commands, and messages.
10. Exclude	Displays the View - Exclude Request pop-up used to specify the exclude options and string.

## Source View Screen

### Note:

A program first displays in the Source View format. Select View ► Source or type SOURCEVIEW to return to Source View from the Tree View or the Perform Structure View.

To display the COBOL source program code in a textual format, follow this step:

- Select View ► Source and press Enter. The Source View screen, shown in [Figure 19](#), displays. This screen displays each source line and functions similar to the ISPF Browse option.

**Figure 19 • Source View Screen**

```

File View Search Logic Task List Options Help
-----
Source View                               Program: VIAIDEMO
Command ==>                               Scroll ==> CSR

***** ***** TOP OF DATA*****
000001 IDENTIFICATION DIVISION.
000002 PROGRAM-ID. VIAIDEMO.
000003 AUTHOR. WRITTEN BY ASG AT LANGLVL 2.
000004 ENVIRONMENT DIVISION.
000005 CONFIGURATION SECTION.
000006 SOURCE-COMPUTER. IBM-370.
000007 OBJECT-COMPUTER. IBM-370.
000008 INPUT-OUTPUT SECTION.
000009 FILE-CONTROL.
000010     SELECT MASTERIN ASSIGN TO S-MASTERIN.
000011     SELECT MASTER-RPT ASSIGN TO S-MREPORT.
000012 DATA DIVISION.
000013 FILE SECTION.
000014 FD MASTERIN
000015     RECORDING MODE IS F
000016     BLOCK CONTAINS 0 RECORDS
000017     LABEL RECORDS ARE STANDARD.
000018
000019     COPY VIAIMAST.

```

The Source View screen displays individual source lines, with the copy members properly expanded. The name of the program that is being viewed displays in the upper right corner of the screen.

To close the active program and display the primary Insight screen, follow this step:

- Press PF3 or type END on the Source View screen. See the ["END Command" on page 200](#) for more information.

## Usage Notes

The Source View screen is also available in the Task facility.

## View - Tree View Request Pop-up

### To format a program to display on the Tree View screen

- 1 Select View ▶ Tree and press Enter. The View - Tree View Request pop-up, shown in [Figure 20](#), displays.

Figure 20 • View - Tree View Request Pop-up

```
View - Tree View Request

Type request options. Then press Enter. For a name selection list
for a range (other than 1 or 2), type a pattern (e.g. ABC*) in the
name field.

Levels 2 of 5 of depth

Contents                                Range
1  1. Labels                             1  1. Procedure division
   2. Labels and performs                 2. All entry points
   3. Labels, performs and GOTOS          3. Perfrange name
   4. Labels, performs, GOTOS and        4. Entry point name
      conditionals                        5. Nested program name

Name _____

Options
- Repetition
```

- 2 Complete these fields:
  - a Specify the number of levels you want to view.  
You can change the levels at any time by selecting View ▶ Levels action or by typing LEVELS and pressing Enter.
  - b Select the type of statements to be displayed in the Contents field.
  - c Specify how much of the program to display in the Range field.

**Note:** \_\_\_\_\_

Each configuration parameter field contains default values when this pop-up first displays. If you change these defaults, the new values are retained for subsequent sessions.

- 3 Press Enter to display the Tree View screen shown in [Figure 21 on page 33](#).

## Fields

Field	Description
Levels	<p>Optional. Specifies the number of hierarchical levels of COBOL source statements that display. Change the level depth value to either increase or decrease the amount of detail displayed. The amount of detail increases as the level depth increases.</p> <p>The total number of levels in the program displays to the right of this field. Type a number equal to or less than this number, or enter the value MAX. When you specify MAX levels, the actual number of levels displayed depends on the Repetition Option. The default is 2.</p>
Contents	<p>Optional. Specifies the number of the statement type(s) to be displayed. Each statement type is a subset of the COBOL statements contained in the program. Insight classifies COBOL statements into subsets by grouping together COBOL verbs of a similar nature.</p> <p>These are the subsets available on this screen:</p> <p>Labels            The PROCEDURE DIVISION statement, and all section and paragraph names in the PROCEDURE DIVISION. This is the default.</p> <p>Performs        Statements containing the PERFORM, SORT, or MERGE verbs.</p> <p>GOTOs            Statements containing an ALTER or GO TO verb.</p> <p>Conditionals    Statements or parts of statements that conditionally change the flow of control in a program such as IF, ELSE, and WHEN.</p>
Range	<p>Optional. Specifies how much of the program is to be displayed.</p> <p>These ranges are available:</p> <p>1. Procedure Division    Displays the entire PROCEDURE DIVISION. This is the default.</p> <p>2. All Entry Points        Displays the entire PROCEDURE DIVISION and all ENTRY points.</p> <p>3. Perfrange Name        Displays one perform range. If you select this range, you must enter the perform range name in the Name field.</p> <p>4. Entry Point Name       Displays one ENTRY point. If you select this range, you must enter the entry point name in the Name field.</p>

Field	Description
	5. Nested Program Name Displays one nested program. If you select this range, you must enter the program name in the Name field of the code that is executed by entering the specified program. Requires Name field.
Name	<p>Specifies the name of the perform range, entry point, or nested program to be displayed. This field is required if you select the Perfrange name, Entry point name, or Nested program name for the Range. This field is ignored if you select either range 1 or 2.</p> <p>If this field is left blank, a selection list pop-up displays allowing you to select the desired name. You can use wildcard characters [question mark (?) for 1 character; asterisk (*) for zero or more characters] in this field to reduce the size of the selection list. On the selection list pop-up, specify the Name by typing S next to the desired name.</p>
Repetition	<p>Determines whether perform range structures entirely display wherever they appear in the program.</p> <p>If not specified, the complete perform range structure displays the first time it displays in the program. Subsequent references to the perform range do not display, and are tagged as REPEAT to the right of the PERFORM statement.</p> <p>If specified, the perform range is fully displayed wherever it occurs. The default is display the perform range only the first time it displays.</p>

### Usage Notes

When you request MAX level depth, the actual number of levels displayed may be different, depending on the value in the Repetition field. MAX displays all levels in the program; however, if the maximum level consists of only repeated perform ranges, then the level display is truncated to the next higher level that does not consist exclusively of repeated perform ranges.

## Tree View Screen

The Tree View screen, shown in [Figure 21](#), displays logical program units and the relationships between them. A logical program unit is a PERFORMed range of code including GO TO code, or a CALLED program. COBOL source lines show Logical programs, and indents show unit relationships.

**Figure 21 • Tree View Screen**

```

File View Search Logic Task List Options Help
-----
                                Tree View                                LINE 1 OF 98
Command ==> _____ Scroll ==> CSR
ASG1582I 5 OF 5 LEVELS DISPLAYED IN THE VIEW.
***** ***** TOP OF DATA *****
000000 PROCEDURE DIVISION.
000001     PERFORM PROGRAM-INIT.
000002     PROGRAM-INIT.
000002     PERFORM P005-VAL-PARM THRU P005-EXIT.
000003     P005-VAL-PARM.
000003     IF DBA-DEPT-CODE > 24                                COND
000003     : PERFORM P999-ABEND-PROGRAM.
000004     : P999-ABEND-PROGRAM.
000004     : CALL 'ABENDPGM' USING ABEND-CODE.                    PGM EXIT
000003     IF DBA-DEPT-CODE < 16                                COND
000003     : PERFORM P999-ABEND-PROGRAM.
000004     : P999-ABEND-PROGRAM.                                REPEATED
000004     : CALL 'ABENDPGM' USING ABEND-CODE.
000003     P005-EXIT.
000002     PERFORM P010-OPEN THRU P019-EXIT.
000003     P010-OPEN.
000003     CALL 'DBAOPEN1' USING DBA-DEPT-CODE.
000003     CALL 'DBAOPEN2' USING DBA-DEPT-CODE.
000003     P019-EXIT.

```

## Tree View Generated Tags

The left column on the Tree View screen indicates the nesting level for each statement. This table lists the valid tags:

Tag	Description
+	Indicates lower levels of program structure not displayed. To view the lower level structure, use the ZOOMIN command.
:	Indicates that the statement or label is artificially indented to highlight a conditional statement.
COND	Indicates the control flow verb is part of a conditional (IF, AT END, etc.).
PGM EXIT	Indicates a program exit.
RECUR	Indicates the PERFORM is recursive.

Tag	Description
REPEATED REP	If a PERFORM range is repeated, a REPEATED tag is placed on the line of the duplicate PERFORM range.
SOURCE SRC	The displayed line is actual source code.

The most basic Tree View displays paragraph and section labels. All executed PERFORM statement labels under the paragraph label containing the PERFORM are indented. Program exits also display.

You can view Control flow statements to understand how the labels are executed. Indentation is directly related to the nesting level of the PERFORM statement.

### **Primary Commands**

These primary commands perform the specified functions on the Tree View screen when they are entered in the command input area:

Command	Description
BRANCH	Locates the occurrence of the PERFORM within the Tree View display. If the label is not shown, a message displays.
END	Exits the Tree View screen.
JUMP	Returns to the Source View screen at the location corresponding to the current cursor location where this command is issued on the Tree View screen. Jump also functions as the J line command.
LEVELS <i>n</i>	Redisplays the Tree View screen to the <i>n</i> th level. Any level less than <i>n</i> is expanded to the requested level. Levels greater than <i>n</i> are removed. The LEVELS command works globally throughout the displayed program.
RESET	Removes highlighting, erases tags in columns 73 through 80, and redisplays excluded lines.

Command	Description
ZOOMIN	<p>Expands the Tree View diagram to the next lower hierarchical level. Typing ZOOMIN on a label expands all PERFORMs contained by the paragraph. If there are no more hierarchical levels, or they have already been expanded, the source displays for that label.</p> <p>If all levels have been displayed and you type ZOOMIN, an error message displays because ZOOMIN cannot be entered on a source line.</p>
ZOOMOUT	<p>Removes levels. The first level removed is the source. If there is no source displayed, lower levels are removed. If both source and lower levels have been removed, no more levels are removed.</p> <p>Paragraphs or PERFORMs are marked as containing results (i.e., they are tagged as USE, MOD, or REF). If the results at the source level are desired, ZOOMIN may be used to see the source and the actual results. Results do not display if they are not outside Tree View's bounds.</p> <p>These commands work the same way in Tree View as they do in Source View:</p> <ul style="list-style-type: none"> <li>• REPEAT</li> <li>• RECALL</li> </ul> <p>The EXCLUDE, FINDXTND, and HIGH commands function the same way in Tree View as they do in Source View except these operands are not applicable in Tree View:</p> <ul style="list-style-type: none"> <li>• FIRST</li> <li>• LAST</li> <li>• PREV</li> <li>• NEXT</li> </ul> <p>The LPRINT and LPUNCH commands can be executed in the Tree View screen, however, the only valid operands are asterisk (*) and NOTE. LPRINT and LPUNCH copy the specified target to the List and Punch file, respectively.</p> <p>The SCROLL command can be executed in the Tree View screen, however, these operands are the only valid operands in Tree View:</p> <ul style="list-style-type: none"> <li>• HI</li> <li>• FIRST</li> <li>• LAST -NHI</li> <li>• NEXT</li> <li>• PREV</li> </ul>

## Line Commands

These line commands perform the specified functions on the Tree View screen:

Line Command	Description
$F_n$	Redisplays $n$ excluded lines beginning with the first excluded line. The default is 1.
J	Returns to the Source View screen at the location where this command is issued on the Tree View screen. This command also functions as the JUMP primary command.
$L_n$	Redisplays $n$ lines starting with the last line in the block of excluded lines.
$S_n$	Redisplays $n$ lines starting with the first line in the block of excluded lines. The default is 1.
SS	Redisplays a block of lines. Type on the first and last line in the block of lines to be redisplayed.
$X_n$	Excludes $n$ lines from being displayed. The default is 1.
XX	Excludes a block of lines from being displayed. Type this command on the first and last line to excluded.
ZI	Performs the ZOOMIN command at the line where it is entered. See <a href="#">"ZOOMIN/ZOOMOUT Commands" on page 324</a> for more information.
ZO	Performs the ZOOMOUT command at the line where it is entered. See <a href="#">"ZOOMIN/ZOOMOUT Commands" on page 324</a> for more information.

## View - Structure View Request Pop-up

*To format the program on the Perform Structure View screen*

- 1 Select View ▶ Structure and press Enter. The View - Structure View Request pop-up, shown in [Figure 22](#), displays.

**Figure 22 • View - Structure View Request Pop-up**

View - Structure View Request

Type request options. Then press Enter. For a name selection list for a range (other than 1 or 2), type a pattern (e.g. ABC\*) in the name field.

Levels 2 of 4 of depth

Range

1 1. Procedure division  
 2. All entry points  
 3. Perform range name  
 4. Entry point name  
 5. Nested program name

Name \_\_\_\_\_

Options  
 - Repetition

- 2 Specify these configuration parameters:
  - a Specify the number of hierarchical levels you want to view.  
 You can change the levels at any time by selecting View ▶ Levels action or by typing LEVELS and pressing Enter.
  - b Specify how much of the program you want to display in the Range field.

**Note:** \_\_\_\_\_

Each field contains default values for the configuration parameters when the View - Structure View Request pop-up first displays. If you change these defaults, the new values are retained for subsequent sessions.

- 3 Press Enter to display the Perform Structure View screen, shown in [Figure 23 on page 40](#).

## Fields

Field	Description
Levels	<p>Specifies the number of hierarchical levels of COBOL source statements that display. Change the level depth value to increase or to decrease the amount of detail displayed. The amount of detail increases as the level depth increases.</p> <p>The total number of levels in the program displays to the right of this field. Type a number equal to or less than this number, or enter the value MAX. When you specify MAX levels, the actual number of levels displayed depends on the Repetition Option. The default is 2.</p>
Range	<p>Specifies how much of the program is to be displayed. These are the valid ranges:</p> <ol style="list-style-type: none"><li>1. Procedure Division      Displays the entire PROCEDURE DIVISION. This is the default.</li><li>2. All Entry Points        Displays the entire PROCEDURE DIVISION and all ENTRY points.</li><li>3. Perfrange Name        Displays one perform range. If you select this range, you must enter the perform range name in the Name field.</li><li>4. Entry Point Name      Displays one ENTRY point. If you select this range, you must enter the entry point name in the Name field.</li><li>5. Nested Program Name    Displays one nested program. If you select this range, you must enter the program name in the Name field of the code that is executed by entering the specified program. Requires the Name field.</li></ol>

Field	Description
Name	<p>Required if you select the Perform range name, entry point name, or nested program name for the Range. This field is ignored if you select either Range 1 or 2.</p> <p>Type the name of the perform range, entry point, or nested subprogram to be displayed.</p> <p>If you leave this field blank, a selection list pop-up displays allowing you to select the desired target name. Wildcard characters [question mark (?) for 1 character; asterisk (*) for zero or more characters] can be entered in this field to reduce the size of the selection list.</p> <p>On the selection list pop-up, specify the Name by typing S next to the desired name.</p>
Repetition	<p>Specifies whether perform range structures display entirely display wherever they appear in the program.</p> <p>If not specified, the complete perform range structure displays the first time it displays in the program. A box containing the notation REPEAT represents subsequent references to the perform range.</p> <p>If specified, the perform range structure fully displays whenever it occurs. The default is to display the perform range only the first time it displays.</p>

### Usage Notes

When you choose MAX level depth, the actual number of levels displayed may be different, depending on the value in the Repetition field. MAX displays all levels in the program; however, if the maximum level consists of only repeated perform ranges, then the level display is truncated to the next higher level that does not consist exclusively of repeated perform ranges.

## Perform Structure View Screen

The Perform Structure View screen, shown in [Figure 23](#), represents a COBOL program structure graphically and reflects the hierarchical relationship between the logical components (usually PERFORMed code and CALLEd programs).

**Figure 23 • Perform Structure View Screen**

```

File View Search Logic List Utility Help
-----
                        Perform Structure View                PROGRAM: VIAIDEMO
====>                                                           SCROLL ==> CSR
ASG1582I 2 OF 4 LEVELS DISPLAYED IN THE VIEW.
+---- 001+
|VIAIDEMO|
|      |
|      |
+-----+
|-----|
+---- 002+                               +---- 003+
|PROGRAM-|                               |P000-NEX|
|INIT    |                               |T THRU P|
|      |                               |000-EXIT|
+-----+                               +-----+
|-----|                               |-----|
+---- 004+ +---- 005+ +---- 006+ +---- 007+ +---- 008+ +---- 009+ +---- 010+
|P005-VAL| |P010-OPE| |P155-CL-| |P120-REA| |'VIAIDEM| |P100-PRI| |P120-REA|
|-PARM TH| |N THRU P| |SUBTOT T| |D THRU P| |1'      | |NT THRU | |D THRU P|
|RU P005*| |019-EXIT| |HRU P15*| |129-EXIT| |      | |P119-EX*| |129-(7)|
+-----+ +-----+ +-----+ +-----+ +-----+ +-----+ +REPEATED+
***** BOTTOM OF DATA *****

```

The Perform Structure View screen displays the logical program units and all transfers of control between these components. Transfers of control include PERFORMs, GOTOs, and CALLs.

Boxes represent logical program units, and lines between the boxes represent relationships. Insight assigns a unique number to each box and displays that number in the upper right corner of the box. The unit name displays in the box. If the name does not fit inside the box, the name is truncated and an asterisk (\*) represents the remaining characters.

If the box contains REPEATED, the first occurrence box number that displays the entire perform range structure is contained inside the box in the lower right corner.

When this screen first displays, a message stating the number of levels displayed, and the total number of program levels displays. Use the Levels option to change the number of levels displayed.

## Usage Notes

Use scrolling commands (UP, DOWN, LEFT, and RIGHT) to see the whole structure chart. The More field displays in the upper right corner of the screen below the SCROLL field when the screen is scrollable. The More field lists the available scrolling directions for the current window, and indicate you can view additional data by scrolling in the indicated direction(s). These are the indicators:

Indicator	Scroll view...
+	DOWN
-	UP
>	RIGHT
<	LEFT

To view the COBOL source code statements for a particular logical program unit box on the Structure View screen, follow this step:

- ▶ Select View ▶ Zoom in, or type ZOOMIN, and press Enter.

The Source View screen displays as a pop-up with a shortened action bar. This screen displays the highlighted COBOL source statements in the requested perform range and excludes all other lines from the display.

**Note:** \_\_\_\_\_

For more information about the ZOOMIN/ZOOMOUT commands, see ["Commands" on page 177](#). For more information about the Zoom in action on the View pull-down, see ["Actions" on page 27](#).

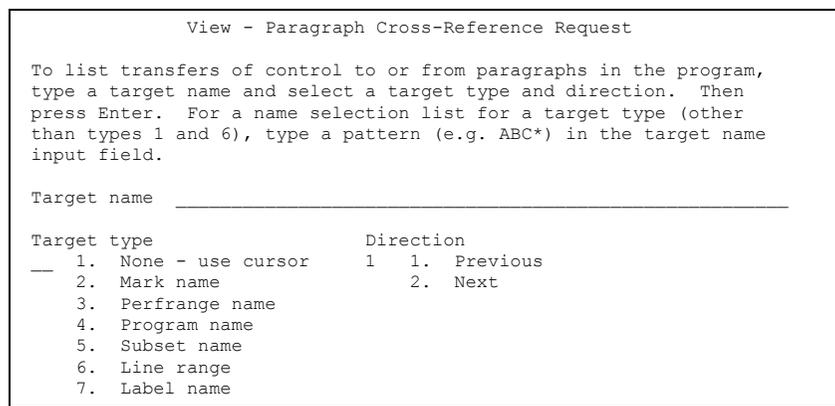
\_\_\_\_\_

## View - Paragraph Cross-Reference Request Pop-up

To customize the View - Paragraph Cross Reference Request pop-up, follow this step:

- ▶ Select View ▶ Paragraph X-ref and press Enter. The View - Paragraph Cross-Reference Request pop-up, shown in [Figure 24](#), displays.

**Figure 24 • View - Paragraph Cross-Reference Request Pop-up**



### Fields

Target Type	Description
Target name	<p>Optional. Type the mark name, perform range name, program name, subset name, line range, or label name used as a basis for the paragraph cross-reference. If you want to use cursor positioning to select the target, leave this field blank. The information you enter in the Target name field must correspond to the information in the Target type field.</p> <p>If this field is left blank and you choose a Target type other than None, a selection list pop-up displays that allows you to select the desired target name. Wildcard characters [question mark (?) for 1 character; asterisk (*) for zero or more characters] can be entered in this field to reduce the size of the selection list. On the selection list pop-up, specify the Name by typing S next to the desired name.</p>
Target type	<p>Optional. Select the type of target represented by the entry in the Target name field. The default is None - use cursor.</p> <p>Choose from these options:</p> <ol style="list-style-type: none"> <li>1. None - Use Cursor Specifies the target by positioning the cursor on the desired target. When you select this Target type, the previous View screen displays to allow cursor placement.</li> </ol>

Target Type	Description
2. Mark	Specifies that the entry in the Target name field is a mark name (i.e., a set of source lines created using the scratchpad pop-ups or the COPY, MARK, MERGE, or RENAME commands). The View - Paragraph Cross Reference pop-up displays all paragraph names that contain the specified mark name.
3. Perfrange	Specifies that the entry in the Target name field is a perform range name. All paragraph names that contain the perform range display on the View - Paragraph Cross Reference pop-up.
4. Program	Specifies that the entry in the Target name field is a program name. All paragraph names that contain the program display on the View - Paragraph Cross Reference pop-up.
5. Subset	Specifies that the entry in the Target name field is a predefined COBOL language subset.

Target Type	Description
-------------	-------------

These are the predefined subsets:

ASsignment	DEBug	IO
CALL	DEFinition	LABel
CIcs	DIRective	MAINline
COBOLII	DIVision	MATH
COBOL/370	DL/I   DL/1	Output
COMment	DML	PARagraph
CONditional	ENtry	PERform
COPY	EXIt   PGMExit	RETurn
DB2/SQL	FALLthrough	SECTion
DDL	GOto	SORTMerge
DEAD	IDMS	STRucture
DEADCode	INClude	TESTed
DEADData	Input	UNTested

**Note:**

See the *ASG-Insight User's Guide* for a description of each subset and subset type. The TESTed and UNTested subsets are only available if you have applied TCA results. See the *ASG-SmartTest TCA User's Guide* for more information.

A screen subset:

- Highlighted | HI
- NONHighlighted | NHI
- Excluded | X
- NONExcluded | NX

A tagged lines subset of tags appearing in columns 73 through 80.

The View - Paragraph Cross Reference pop-up displays all paragraph names that contain statements of the specified subset type.

Line	Specifies the Target name field entry is a single line number or range of lines. Columns 1 through 6 of the Source View display Line numbers. The View - Paragraph Cross Reference pop-up displays all paragraph names that contain the specified line(s).
------	--

Target Type	Description
Label	Specifies the entry in the Target name field is a label (i.e., any PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC). The View - Paragraph Cross Reference pop-up displays all paragraph names that contain the specified label.
Direction	Specifies the type of transfer. These are the valid values: <ul style="list-style-type: none"> <li>Previous Displays paragraphs where control is transferred to the target paragraph. The default is Previous.</li> <li>Next Displays paragraphs where control is transferred from the target paragraph.</li> </ul>

## View - Paragraph Cross Reference Pop-up

The View - Paragraph Cross Reference pop-up shows how control is transferred to or from the target paragraphs. Use this pop-up to determine these items:

- How to get to a paragraph
- Which paragraphs transfer control to a paragraph
- Which paragraphs does this paragraph transfer control to

To display The View - Paragraph Cross Reference pop-up, follow this step:

- ▶ Type PREF on any screen or press Enter on the View - Paragraph Cross-Reference Request pop-up. The View - Paragraph Cross Reference pop-up, shown in [Figure 25](#), displays.

**Figure 25 • View - Paragraph Cross Reference Pop-up**

```

View - Paragraph Cross Reference
Command ==> _____ Scroll ==> CSR
Target(s): LABEL PROCEDURE
Direction: NEXT

  A : Add to target      P : Execute PREF      S : Select for viewing
S  Target paragraph(s)  How                Goes to
-----
-  PROCEDURE DIVISION  PERFORM            PROGRAM-INIT
-  " "                 PERFORM            P000-NEXT
-  " "                 PROGRAM EXIT
***** BOTTOM OF DATA *****

```

To copy data to the List file, follow this step:

- ▶ Use the LPRINT \* command on this pop-up.

## Fields

Line Command	Description
Target(s)	Specifies the paragraph name(s) of the requested target.
Direction	Specifies the type of transfer. PREVIOUS highlights paragraphs where control is transferred from and NEXT highlights paragraphs where control is transferred to. The default is PREVIOUS.
Line command area	<p>Specifies the area to the left of the Comes from or Target paragraph(s) field, which accepts these entries:</p> <ul style="list-style-type: none"> <li>A Adds the selected paragraph to the target paragraph(s). If the direction is PREVIOUS, the target paragraph is added at the top of the list. If the direction is NEXT, the target paragraph is added to the end of the list.</li> <li>P Executes another paragraph cross-reference.</li> <li>S Returns to the source code with the first line of the paragraph displayed at the top of the screen.</li> </ul>
Target Paragraph(s)	The paragraphs requested as the target in the PREF command. Ditto marks (") display when multiple paragraphs transfer control to the same paragraph.
How	<p>Indicates the way control is transferred from or to the target paragraph.</p> <p>If the target is reached through more than one occurrence of the same transfer method, the number of occurrences, in parentheses, follows the method of transfer. For example:</p> <p style="padding-left: 40px;">PERF(2), GO(3)</p> <p>indicates that the target was reached through 2 PERFORMs and 3 GOTOS.</p> <p><b>Note:</b> _____</p> <p>For a complete list of paragraph control transfer methods and abbreviations, see "<a href="#">Paragraph Control Transfer Methods</a>" on <a href="#">page 383</a>.</p> <p>_____</p>
Comes from	Displays when the direction is PREVIOUS. It lists the paragraph names that transfer control to the target paragraph(s). All paragraphs that relate to the target paragraph are grouped together.
Goes to	Displays when the direction is NEXT. It lists the paragraph names that receive control from the target paragraph(s).

## View - Program Summary Pop-up

*To select the type of summary information to view*

- 1 Select View ► Program Summary and press Enter. The View - Program Summary pop-up, shown in [Figure 26](#), displays.

**Figure 26 • View - Program Summary Pop-up**

```

View - Program Summary

Options
1  1.  Files
    2.  Data
    3.  Copy members
    4.  Sections
    5.  Control flow
    6.  CICS, DB2, etc.

```

- 2 Type the number corresponding to the type of summary information you want to view.

Each option on the View - Program Summary pop-up presents a collection of facts about a particular program entity type, such as a file, data item, perform range, or called subprogram. The default is Files. View the program summary information to save time and facilitate decision making.

### Fields

Option	Description
Files	Displays the Program Summary - Files pop-up showing counts of file usage and statements.
Data	Displays the Program Summary - Data pop-up, that shows the number of statements that include the various types of uses of data items.
Copy members	Displays the Program Summary - Copy Members pop-up, indicating the number of lines copied for all copy members.
Sections	Displays the Program Summary - Sections/Paragraphs pop-up, that shows the number of lines in the various program sections and paragraphs.

Option	Description
Control flow	Displays the Program Summary - Control Flow pop-up, indicating the number of each type of control flow statement.
CICS, DB2, etc.	Displays the Program Summary - Preprocessor Statements pop-up indicating the number of database statements, including SQL, IDMS, CICS, and DL/I

## Program Summary - Files Pop-up

To display file usage and statement counts, follow this step:

- ▶ Select View ▶ Program Summary ▶ Files and press Enter. The Program Summary - Files pop-up, shown in [Figure 27](#), displays.

**Figure 27 • Program Summary Files Pop-up**

Program Summary - Files				
	FD Seq	FD Rel	FD Indx	SD (Seq)
	-----	-----	-----	-----
Usage:				
Input only . . . :	1	0	0	0
Output only . . . :	2	0	0	0
Input/Output . . . :	0	0	0	0
Total File Count . . . :	3	0	0	0
Input:				
Statements . . . :	1	0	0	0
Paragraphs . . . :	1	0	0	0
Output:				
Statements . . . :	10	0	0	0
Paragraphs . . . :	5	0	0	0

## Fields

Field	Description
FD Seq	Specifies File Descriptions (FDs) defined with ORGANIZATION IS SEQUENTIAL in the FILE-CONTROL paragraph.
RD Rel	Specifies File Descriptions (FDs) defined with ORGANIZATION IS RELATIVE in the FILE-CONTROL paragraph.
FD Indx	Specifies File Descriptions (FDs) defined with ORGANIZATION IS INDEXED in the FILE-CONTROL paragraph.
SD (Seq)	Specifies Sort File Descriptions (SDs), categorized by Insight as sequential files.

Field	Description
Input only	Specifies the number of FD and SD definitions referenced only in input (e.g., READ) statements.
Output only	Specifies the number of FD and SD definitions referenced only in output (e.g., WRITE) statements.
Input/Output	Specifies the number of FD and SD definitions referenced in both input (e.g., READ) and output (e.g., WRITE, REWRITE) statements.
Total File Count	Specifies the number of FD and SD definitions in the FILE SECTION.
Input Statements	Specifies the number of statements that reference an FD or SD definition.
Paragraphs	Specifies the number of paragraphs containing input statements that reference an FD or SD definition.
Output Statements	Specifies the number of statements that reference an FD or SD definition.
Paragraphs	Specifies the number of paragraphs containing output statements that reference an FD or SD definition.

## Program Summary - Data Pop-up

To display the number of statements that include the various data item types in the program, follow this step:

- ▶ Select View ▶ Program Summary ▶ Data and press Enter. The Program Summary - Data pop-up, shown in [Figure 28](#), displays.

**Figure 28 • Program Summary - Data Pop-up**

Program Summary - Data					
	01/77	Subitem	88	COMP	DISPLAY
Total Count . . . . :	24	133	1	10	0
Usage:					
Redefines . . . . . :	1	0	0	0	0
Modified, not used	7	81	0	2	0
Used, not modified	7	22	1	1	0
Indirect ref. only	4	19	0	0	0
Dead . . . . . :	3	4	0	2	0
Input statements :	1	0	0	0	0
Output statements :	10	0	0	1	0
Role:					
SQL host variable :	0	0	0	0	0
Linkage . . . . . :	1	3	0	1	0
CALL ... Using . . . :	3	2	0	1	0
in FD/RD/SD . . . . :	2	24	0	0	0

### Fields

Field	Description
01/77	Specifies the number of data variables defined as 01 or 77 level data items.
Subitem	Specifies the number of data variables defined as subordinate to 01 level data items, with the exception of 88 level data items that are counted separately.
88	Specifies the number of data variables defined as 88 level data items.
COMP	Specifies the number of data items defined with USAGE IS COMP.
DISPLAY	Specifies the number of data items defined with USAGE IS DISPLAY.
Total Count	Specifies the total number of data items defined as 01/77, Subitem, 88, COMP, and DISPLAY.
Redefines	Specifies the number of data items defined with a REDEFINES clause.
Modified, not used	Specifies the number of data items modified but not used.



## Fields

Field	Description
Member Name	Specifies the member name as it displays in the copy statement.
Lines	Specifies the number of source lines copied for that member.
Contents	<p>Describes the overall contents of the copied member. These are the valid descriptions:</p> <ul style="list-style-type: none"> <li>• IDENTIFICATION DIVISION lines</li> <li>• ENVIRONMENT DIVISION lines</li> <li>• FD, SD, RD, and CD definitions</li> <li>• 01, 88, and 77 level items</li> <li>• OTHER level items (i.e., 02 through 49 level)</li> <li>• PROCEDURE DIVISION statements</li> <li>• Program elements from multiple DIVISIONs</li> </ul>

## Program Summary - Sections/Paragraphs Pop-up

To show the number of lines in the various program sections and paragraphs, follow this step:

- ▶ Select View ▶ Program Summary ▶ Sections and press Enter. The Program Summary - Sections/Paragraphs pop-up, shown in [Figure 30](#), displays.

**Figure 30 • Program Summary - Sections/Paragraphs Pop-up**

```

Program Summary - Sections/Paragraphs

Location                Lines
-----
Data Division . . . . . : 219
  File Section . . . . . : 39
  Working-Storage Section : 171
  Linkage Section . . . . : 8
Procedure Division . . . : 266
  Number of SECTIONS . . : 1
  Number of labels . . . : 26
    "EXIT only" . . . . : 10
    GOTO/ALTER targets . : 4
    In-line code . . . . : 4
    PERFORMed . . . . . : 22
    Deadcode . . . . . : 4
  Number of statements . : 110
    Within PERFORMs . . . : 99
    Deadcode . . . . . : 4
Total . . . . . : 496
    
```

## Fields

Field	Description
Data Division	<p>Specifies the total number of source lines, including comments, for each of these sections:</p> <ul style="list-style-type: none"> <li>• File Section</li> <li>• Working-Storage Section</li> <li>• Linkage Section</li> </ul>
Procedure Division	<p>Specifies the total number of PROCEDURE DIVISION source lines, including comments. These subtotals are included:</p> <ul style="list-style-type: none"> <li>• The number of SECTIONS in the PROCEDURE DIVISION.</li> <li>• The number of paragraphs in the PROCEDURE DIVISION, including these areas: <ul style="list-style-type: none"> <li>— the number of paragraphs containing only an EXIT statement</li> <li>— and the number of paragraphs targeted by GO TO or ALTER statements.</li> <li>— The number of paragraphs not reached via PERFORMs (i.e., they execute as a result of fall through code).</li> <li>— The number of paragraphs reached via PERFORMs.</li> <li>— The number of paragraphs containing dead code (i.e., code that is never reachable).</li> </ul> </li> </ul> <p>The number of statements (not source lines) in the PROCEDURE DIVISION are included for these statements:</p> <ul style="list-style-type: none"> <li>• The number of statements executed by PERFORM statements.</li> <li>• The number of unreachable statements.</li> </ul>
Total	<p>Specifies the number of source lines in the program, including comment lines.</p>

## Program Summary - Control Flow Pop-up

To show the number of each control flow statement type in the program, follow this step:

- ▶ Select View ▶ Program Summary ▶ Control flow and press Enter. The Program Summary - Control Flow pop-up, shown in [Figure 31](#), displays.

Figure 31 • Program Summary - Control Flow Pop-up

	Count
External CALL statements	0
Programs CALLED	0
PERFORM statements	0
Number of PERFORM ranges	0
Ranges PERFORMed more than once	0
GO TO statements	0
Number of labels reached	0
Labels reached more than once	0
Number of ALTERed GO TOs	0
Multi-way branches	0
GO TO...DEPENDING ON...	0
EVALUATE...	0
ON CONDITION statements	0
End-of-file (AT END...)	0
Other Input/Output conditions	0

### Fields

Field	Description
External CALL statements	Specifies the number of CALL statements in the PROCEDURE DIVISION. A subtotal of the number of unique programs called is included.
PERFORM statements	Specifies the number of PERFORM statements in the program. Subtotals are included for the number of unique Perform Range targets and the number of ranges performed multiple times.
GO TO statements	Specifies the number of GO TO statements in the program. Subtotals are included for the number of unique labels that are targets of GO TOs, the number of labels that can be reached from multiple GO TOs, and the number of GO TOs that are the targets of ALTER statements.

Field	Description
Multi-way branches	Specifies the number of GO TO DEPENDING ON and EVALUATE statements in the program. Subtotals are included for the number of GO TO DEPENDING ON and EVALUATE statements.
ON CONDITION statements	Specifies the number of statements containing exception condition clauses, including SIZE, OVERFLOW, INVALID KEY, NO DATA, END OF PAGE, and END OF FILE. Subtotals are included for the number of AT END condition statements, the number of other I/O exceptions (e.g., INVALID KEY), and the number of arithmetic exceptions (e.g., ON OVERFLOW, ON SIZE ERROR, etc.).

## Program Summary - Preprocessor Statements Pop-up

To display the number of database statements in the program, including SQL, IDMS, CICS, and DL/I, follow this step:

- ▶ Select View ▶ Program Summary ▶ CICS, DB2, etc. The Program Summary - Preprocessor Statements pop-up, shown in [Figure 32](#), displays.

**Figure 32 • Program Summary - Preprocessor Statements Pop-up**

Program Summary - Preprocessor Statements				
	SQL	IDMS	CICS	DL/I
	----	----	----	----
Usage:				
Input only . . . :	7	0	N/A	N/A
Output only . . . :	1	0	N/A	N/A
Input/Output . . . :	2	0	N/A	N/A
Total Count . . . :	10	0	N/A	N/A
Input:				
Statements . . . :	2	0	0	0
Paragraphs . . . :	2	0	0	0
Output:				
Statements . . . :	6	0	0	0
Paragraphs . . . :	5	0	0	0

**Fields**

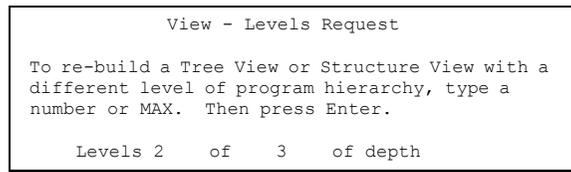
Field	Description
SQL	Specifies the number of Tables and Views referenced in EXEC SQL statements.
IDMS	Specifies the number of IDMS Record Names referenced in IDMS COBOL statements.
CICS	Specifies the number of EXEC CICS statements.
DL/I	Specifies the number of EXEC DL/I statements, including CALL 'CBLTDLI' statements.
Input only	Specifies the number of database definitions referenced only by statements that query the database.
Output only	Specifies the number of database definitions referenced only by statements that update the database.
Input/Output	Specifies the number of database definitions referenced by statements that query or update the database.
Total Count	Specifies the number of database definitions.
Input Statements	Specifies the number of query-type statements that reference a database definition.
Input Paragraphs	Specifies the number of paragraphs containing query-type statements that reference a database definition.
Output Statements	Specifies the number of update-type statements that reference a database definition.
Output Paragraphs	Specifies the number of paragraphs containing update-type statements that reference a database definition.

## View - Levels Request Pop-up

To redisplay the current Tree View or Perform Structure View screen to the specified level, follow this step:

- ▶ Select View ▶ Levels and press Enter. The View - Levels Request pop-up, shown in [Figure 33](#), displays.

**Figure 33 • View - Levels Request Pop-up**



### Field

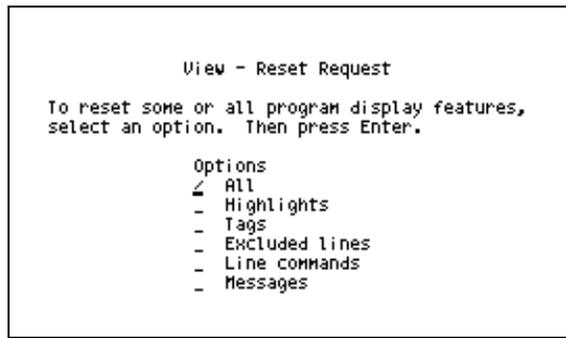
Field	Description
Levels	<p>Specifies the number of hierarchical levels to be displayed. Change the level depth value to increase or decrease the amount of detail displayed. The amount of detail increases as the level depth increases.</p> <p>The total number of levels in the program displays to the right of this field. Type a number equal to or less than this number, or type the value MAX to display all levels. The default is 2.</p>

## View - Reset Request Pop-up

To remove highlighting, tags, messages, redisplay excluded lines, and cancel pending line commands, follow this step:

- ▶ Select View ▶ Reset and press Enter. The View - Reset Request pop-up, shown in [Figure 34](#), displays.

Figure 34 • View - Reset Request Pop-up



### Fields

Field	Description
All	Optional. If specified, all conditions indicated by each of the other fields on this pop-up reset. If not specified, only those options selected reset. The default is to reset all conditions.
Highlights	Optional. If specified, any line that was highlighted is returned to nonhighlighted. If not specified, highlighted lines remain highlighted unless the All option is specified. The default is lines remain highlighted.
Tags	Optional. If specified, tags in the columns 73 through 80 are erased. If not specified, tags remain unless the All option is specified. The default is lines remain tagged.
Excluded lines	Optional. If specified, excluded lines redisplay. If not specified, excluded lines remain excluded unless the All option is specified. The default is lines remain excluded.

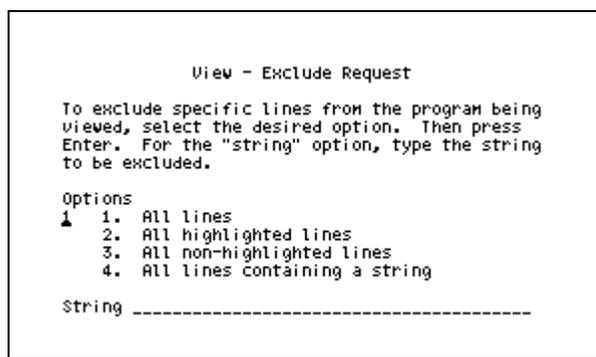
Field	Description
Line commands	Optional. If specified, pending line commands cancel. If not specified, pending line commands remain on the screen unless the All option is specified. The default is pending line commands remain displayed.
Messages	Optional. If specified, short or long messages are removed from the display. If not specified, messages remain displayed on the screen unless the All option is specified. The default is messages remain displayed.

## View - Exclude Request Pop-up

To exclude lines from the screen display, follow this step:

- ▶ Select View ▶ Exclude and press Enter. The View - Exclude Request pop-up, shown in [Figure 35](#), displays.

**Figure 35 • View - Exclude Request Pop-up**



A line of dashes and text stating n LINE(S) NOT DISPLAYED represent excluded lines. The Exclude action is valid on Source View and Tree View.

## Fields

Field	Description
Options	<p>Specifies the lines to exclude from the screen display. These are the valid options:</p> <ol style="list-style-type: none"><li>1. All Lines Excludes all lines from the screen display. This is the default.</li><li>2. All Highlighted Lines Excludes all lines highlighted as the result of one or more actions or commands from the screen display.</li><li>3. All Non-highlighted Lines Excludes all non-highlighted lines from the screen display.</li><li>4. All Lines Containing a String Excludes all lines containing the string specified in the String field. If this Option is selected, the String field is required.</li></ol>
String	<p>Required if Option 4 is selected. Type a string of alphanumeric or DBCS characters. This field is ignored if any other Option is selected.</p>

---

# 4

## Search

---

This chapter describes items on the Insight Search menu and contains these sections:

Topic	Page
<a href="#">Search Actions</a>	<a href="#">62</a>
<a href="#">Search Pull-down</a>	<a href="#">65</a>
<a href="#">Search - Data Name Pop-up</a>	<a href="#">66</a>
<a href="#">Search - Label Name Pop-up</a>	<a href="#">70</a>
<a href="#">Search - Paragraph Name Pop-up</a>	<a href="#">72</a>
<a href="#">Search - Pattern String Pop-up</a>	<a href="#">75</a>
<a href="#">Search - COBOL Subset Name Pop-up</a>	<a href="#">78</a>
<a href="#">Search - Perform Range Name Pop-up</a>	<a href="#">81</a>
<a href="#">Search - Program Name Pop-up</a>	<a href="#">84</a>
<a href="#">Search - User Mark Name Pop-up</a>	<a href="#">86</a>
<a href="#">Search - Line Range Pop-up</a>	<a href="#">88</a>
<a href="#">Search - Any/Unknown Type Pop-up</a>	<a href="#">91</a>
<a href="#">Search - Branch Request Pop-up</a>	<a href="#">93</a>
<a href="#">Branch Previous Options Pop-up</a>	<a href="#">95</a>
<a href="#">Selection List Pop-ups</a>	<a href="#">96</a>
<a href="#">IN-Clause Option Pop-up</a>	<a href="#">97</a>

The Search facility enables you to find, highlight, scroll, print, punch, or exclude a specified target. Depending on the value in the Action field and the current view screen, these items can be a target:

- Dataname
- Label name
- Paragraph name
- Pattern
- COBOL subset
- Perform range
- Program
- User mark
- Line number

The Search facility has three different views to assist you in isolating, understanding, and identifying important sections of the source program. The three views, the Source View, Tree View, and Perform Structure View, are created from the same original COBOL source program, but present a different perspective. These views are concurrently available and synchronized with the other views. You can switch back and forth between the views and retain the results of certain actions.

## Search Actions

### *Find*

Use the Find action to search the source code for one or all occurrences of the target. The results are presented in these views:

View	Description
Source View	Positions the cursor at the first target in the specified direction, highlights all occurrences found, and places tags indicating the type of target found to the right of each statement.
Tree View	Positions the cursor at the first perform range containing the target in a top/down manner, highlights all found occurrences, and places tags indicating the type of target found to the right of each statement.
Perform Structure View	Positions the cursor at the first perform range and highlights all results found.

## Highlight

Use the Highlight action to highlight source code lines or boxes containing the specified target. Previously highlighted lines or boxes remain unchanged. The results are presented in these views:

View	Description
Source View	Positions the cursor at the first target in the specified direction, highlights all occurrences found, and places tags indicating the type of target found to the right of each statement.
Tree View	Positions the cursor at the first perform range containing the target in a top/down manner, highlights all found occurrences, and places tags indicating the type of target found to the right of each statement.
Perform Structure View	Positions the cursor at the first perform range and highlights all results found.

## Scroll

Use the Scroll action to position the screen to the line or box containing the target. Previously highlighted lines or boxes remain unchanged. The results are presented in this manner. The results for this action are the same in each view.

**Note:**

This action is valid only on the Search - Any/Unknown Type pop-up for Tree View and Perform Structure View.

## Print

Use the Print action to copy lines or boxes containing the specified target to the List file for subsequent processing. The results for this Action are the same in each view.

**Note:**

This action is valid only on the Search - Any/Unknown Type pop-up for Tree View and Perform Structure View.

### *Punch*

Use the Punch action to copy lines or boxes containing the target to the Punch file for subsequent processing. The results for this action are the same in each view.

**Note:** \_\_\_\_\_

This Action is valid only on the Search - Any/Unknown Type pop-up for Tree View and Perform Structure View.

---

### *Exclude*

Use the Exclude action to exclude source code lines containing the target. The results are presented in these views:

View	Description
Source View	Searches the source code for one or all occurrences of the target and omits the resulting source code lines.
Tree View	Searches the source code for one or all occurrences of the target and omits the resulting lines.
Perform Structure View	Searches the source code for one or all occurrences of the target and positions the cursor to the first box containing an excluded source code line.

---

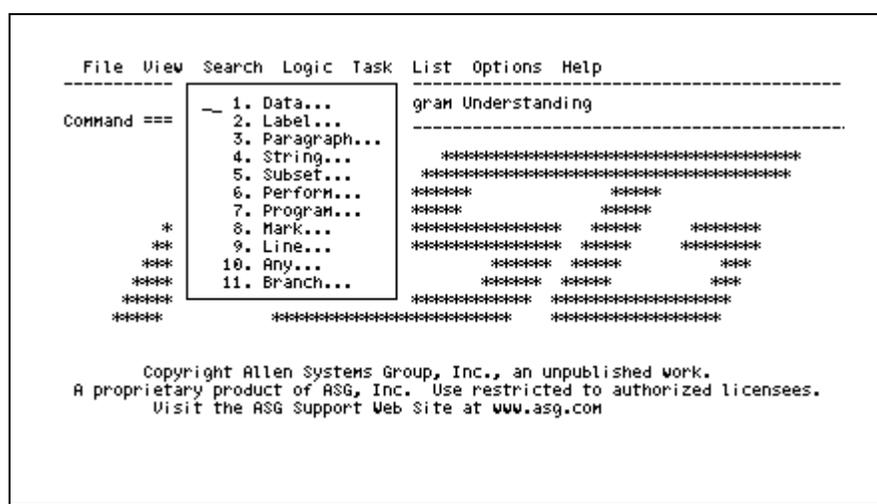
## Search Pull-down

Use the Search pull-down shown in [Figure 36](#) to conduct an intelligent source code search for one or all occurrences of a specified target. This section describes the actions available on the Search pull-down.

To display the Search pull-down, follow this step:

- ▶ Select Search on the action bar and press Enter.

Figure 36 • Search Pull-down



## Actions

Action	Description
1. Data	Displays the Search - Data Name pop-up used to search for data items.
2. Label	Displays the Search - Label Name pop-up used to search for a specified label name.
3. Paragraph	Displays the Search - Paragraph Name pop-up used to search for statements in a paragraph.
4. String	Displays the Search - Pattern String pop-up used to search for a character string.
5. Subset	Displays the Search - COBOL Subset Name pop-up used to search for COBOL subsets.
6. Perform	Displays the Search - Perform Range Name pop-up used to search for statements within a perform range.

Action	Description
7. Program	Displays the Search - Program Name pop-up used to search for statements in subprograms.
8. Mark	Displays the Search - User Mark Name pop-up used to search for statements in a set of marked lines.
9. Line	Displays the Search - Line Range pop-up used to search for statements within line ranges.
10. Any	Displays the Search - Any/Unknown Type pop-up used to search for targets when the type is unknown.
11. Branch	Displays the Search - Branch Request pop-up used to branch to a specific location.

## Search - Data Name Pop-up

To find, highlight, scroll, print, punch, or exclude datanames, depending on the value in the Action field and the current view screen, follow this step:

- ▶ Select Search ▶ Data and press Enter. The Search - Data Name pop-up, shown in [Figure 37](#), displays.

**Figure 37 • Search - Data Name Pop-up**

```

Search - Data Name

Type a data name and select search options. Then press Enter. For
a selection list, enter a pattern (e.g. ABC*) in the name area.

Data name _____

References          Indirect impact          Size change
1  1. All           1  1. None                levels . . . ____
   2. Defs         2. Of size change
   3. Uses         3. Of value change
   4. Mods

Direction          Options                    Action
1  1. All          _ No data aliasing       1  1. Find
   2. Next        _ IN-clause...           2. Highlight
   3. Previous
   4. First
   5. Last
  
```

## Fields

Field	Description
Data name	<p>Specifies a COBOL dataname or qualified COBOL dataname. A dataname refers to any valid COBOL reference for a data element.</p> <p>To display the Selection List pop-up, leave the Data Name field blank or type a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters].</p> <p><b>Note:</b> _____ See "<a href="#">Selection List Pop-ups</a>" on page 96 for more information.</p> <p>Datanames can be concatenated by placing a plus sign (+) between the datanames.</p>
References	<p>Restricts the search to only those datanames that are defined, used, or modified, when you specify the type of dataname reference. These are the valid references:</p> <ol style="list-style-type: none"> <li>1. All Includes occurrences of the dataname where its value is defined, tested, used, set, or modified. This is the default.</li> <li>2. Defs Includes definitions of the dataname in the DATA DIVISION.</li> <li>3. Uses Includes occurrences of the dataname where its value is being tested or used.</li> <li>4. Mods Includes occurrences of the dataname where its value is being set or modified.</li> </ol>
Indirect impact	<p>Expands the search to include occurrences of datanames indirectly affected by the specified dataname. These are the valid values:</p> <ol style="list-style-type: none"> <li>1. None Includes only occurrences of the dataname where it is directly tested, used, set, modified, or defined (based on the value of the References field). This is the default.</li> <li>2. Of size change Includes datanames that could be directly or indirectly affected if the size of the specified dataname were to be changed.</li> <li>3. Of value change Includes datanames that could be directly or indirectly affected if the value of the specified dataname were to be changed.</li> </ol>

Field	Description						
Size change levels	Identifies the depth of the indirect impact of a size change that is desired. When this field is used, all references affected within the specified level show. If left blank, all levels are searched						
Direction	<p>Restricts the search to a specific direction or occurrence. These are the valid values:</p> <ol style="list-style-type: none"> <li>1. All Searches for all occurrences of the dataname and displays the number found in the short/long message field. This is the default.</li> <li>2. Next Searches forward from a specified cursor position to the next occurrence of the dataname. Next is valid on Source View only.</li> <li>3. Previous Searches backward from a specified cursor position to the previous occurrence of the dataname. Previous is valid on Source View only.</li> <li>4. First Searches from the top of the source file to the first occurrence of the dataname. First is valid on Source View only.</li> <li>5. Last Searches backward from the bottom of the source file to the last occurrence of the dataname. Last is valid on Source View only.</li> </ol>						
Options	<p><b>No Data Aliasing</b> Specifies that aliases for the specified dataname are ignored. If left blank, aliases of the dataname are included. This is the default.</p> <p>Aliases include:</p> <table border="0"> <tr> <td>Parent</td> <td>higher level group item</td> </tr> <tr> <td>Child</td> <td>lower level item</td> </tr> <tr> <td>Rename/Redefinition</td> <td>renamed, redefined, or 88 level items</td> </tr> </table> <p><b>IN-clause</b> Optional. If specified, displays the IN-Clause Option pop-up used to restrict the source lines or boxes to be considered for a search. If left blank, the entire source code program is searched. The default is the entire program. See <a href="#">"IN-Clause Option Pop-up" on page 97</a> for more information.</p>	Parent	higher level group item	Child	lower level item	Rename/Redefinition	renamed, redefined, or 88 level items
Parent	higher level group item						
Child	lower level item						
Rename/Redefinition	renamed, redefined, or 88 level items						

Field	Description
Action	<p>Optional. Specify the action to be performed on the dataname. These are the valid values:</p> <p><b>Note:</b> _____  Results depend on the current view screen. See "<a href="#">Search Actions</a>" on <a href="#">page 62</a> for more information.</p> <hr/> <p>1. Find Searches for one or all occurrences of the dataname. Double Byte Character Set (DBCS) identifiers or patterns can be specified to locate DBCS strings. Find is the default.</p> <p>2. Highlight Highlights source code lines or boxes containing the dataname. Lines or boxes already highlighted are not reset.</p> <p>3. Scroll Positions the screen to the first line containing the dataname. Previously highlighted lines remain unchanged. Scroll is valid on Source View only.</p> <p>3. Print Copies lines containing the dataname to the List file. The List file is processed on the Options - Log/List/Punch Definition pop-up. Print is valid on Source View only.</p> <p>4. Punch Copies lines containing the dataname to the Punch file. The Punch file is processed on the Options - Log/List/Punch Definition pop-up. Punch is valid on Source View only.</p> <p>5. Exclude Omits source code lines containing occurrences of the dataname from the display. In Perform Structure View, the cursor is positioned to the first box representing a logical program unit that contains an excluded source code line.</p>

## Search - Label Name Pop-up

Use the Search - Label Name pop-up to find, highlight, scroll, print, punch, or exclude a specified label name and all control transfers to that label name, depending on the value in the Action field and the current view method.

To display the Search - Label Name pop-up, follow this step:

- ▶ Select Search ▶ Label and press Enter. The Search - Label Name pop-up, shown in [Figure 38](#), displays.

**Figure 38 • Search - Label Name Pop-up**

Search - Label Name

To find label name references, type a name and select search options. Then press Enter. For selection list, type a pattern (e.g. ABC\*) in the name area.

Label name \_\_\_\_\_

Direction	Options	Action
1 1. All	_ IN-clause...	1 1. Find
2. Next		2. Highlight
3. Previous		3. Scroll
4. First		4. Print
5. Last		5. Punch
		6. Exclude

### Fields

Field	Description
Label name	<p>Required. Type any PROCEDURE DIVISION paragraph name or section name. You can also specify the PROCEDURE and PROC literals.</p> <p>To display the Selection List pop-up, leave the Label Name field blank, or enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters].</p> <p><b>Note:</b> _____ See <a href="#">"Selection List Pop-ups" on page 96</a> for more information.</p> <p>Label names may be concatenated by placing a plus sign (+) between the label names.</p>

Field	Description
Direction	<p>Restricts the search to a specific direction or occurrence. These are the valid values:</p> <ol style="list-style-type: none"> <li>1. All Searches for all occurrences of the label name and displays the number found in the short/long message field. This is the default.</li> <li>2. Next Searches forward from a specified cursor position to the next occurrence of the label name. Next is valid on Source View only.</li> <li>3. Previous Searches backward from a specified cursor position to the previous occurrence of the label name. Previous is valid on Source View only.</li> <li>4. First Searches from the top of the source file to the first occurrence of the label name. First is valid on Source View only.</li> <li>5. Last Searches backward from the bottom of the source file to the last occurrence of the label name. Last is valid on Source View only.</li> </ol>
IN-clause	<p>Displays the IN-Clause Option pop-up used to restrict the source lines or boxes to be considered for a search. If left blank, the entire source code program is searched. The default is the entire program. See <a href="#">"IN-Clause Option Pop-up" on page 97</a> for more information.</p>
Action	<p>Specifies the action to be performed on the target. These are the valid actions:</p> <p><b>Note:</b> _____ Results depend on the current view screen. See <a href="#">"Search Actions" on page 62</a> for more information</p> <hr/> <ol style="list-style-type: none"> <li>1. Find Searches for one or all occurrences of the label name. This is the default.</li> <li>2. Highlight Highlights source code lines or boxes containing the label name. Lines or boxes already highlighted are not reset.</li> <li>3. Scroll Positions the screen to the first line containing the label name. Previously highlighted lines remain unchanged. Scroll is valid on Source View only.</li> </ol>

Field	Description
4. Print	Copies lines containing the label name to the List file. The List file is processed on the Options - Log/List/Punch Definition pop-up. Print is valid on Source View only.
5. Punch	Copies lines containing the label name to the Punch file. The Punch file is processed on the Options - Log/List/Punch Definition pop-up. Punch is valid on Source View only.
6. Exclude	Omits lines containing occurrences of the label name from the display. In Perform Structure View, the cursor is positioned to the first box representing a logical program unit that contains an excluded source code line.

## Search - Paragraph Name Pop-up

Use the Search - Paragraph Name pop-up to find, highlight, scroll, print, punch, or exclude statements in a specified paragraph, depending on the value in the Action field and the current view method.

To display Search - Paragraph Name pop-up, follow this step:

- ▶ Select Search ▶ Paragraph and press Enter. The Search - Paragraph Name pop-up, shown in [Figure 39](#), displays.

**Figure 39 • Search - Paragraph Name Pop-up**

Search - Paragraph Name

To find all statements in a paragraph, type a name and select the search options. Then press Enter. For a name list, type a pattern (e.g. ABC\*) in the name area.

Paragraph name \_\_\_\_\_

Direction	Options	Action
1 1. All	_ IN-clause...	1 1. Find
2. Next		2. Highlight
3. Previous		3. Scroll
4. First		4. Print
5. Last		5. Punch
		6. Exclude

## Fields

Field	Description
Paragraph Name	<p>Specifies the name of the paragraph to search.</p> <p>To display the Selection List pop-up, leave the Paragraph Name field blank, or enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters].</p> <p><b>Note:</b> _____ See "<a href="#">Selection List Pop-ups</a>" on page 96 for more information.</p> <p>Paragraph names may be concatenated by placing a plus sign (+) between the paragraph names.</p>
Direction	<p>Optional. The search can be restricted to a specific direction or occurrence by specifying a direction. These are the valid values:</p> <ol style="list-style-type: none"> <li>1. All            Searches for all occurrences of the paragraph name and displays the number found in the short/long message field. This is the default.</li> <li>2. Next           Searches forward from a specified cursor position to the next occurrence of the paragraph name. Next is valid on Source View only.</li> <li>3. Previous       Searches backward from a specified cursor position to the previous occurrence of the paragraph name. Previous is valid on Source View only.</li> <li>4. First           Searches from the top of the source file to the first occurrence of the paragraph name. First is valid on Source View only.</li> <li>5. Last            Searches backward from the bottom of the source file to the last occurrence of the paragraph name. Last is valid on Source View only.</li> </ol>
IN-clause	<p>Displays the IN-Clause Option pop-up used to restrict the source lines or boxes to be considered for a search.</p> <p>If left blank, the entire source code program is searched. The default is the entire program.</p> <p>See "<a href="#">IN-Clause Option Pop-up</a>" on page 97 for more information.</p>

Field	Description
Action	<p>Specifies the action to be performed on the paragraph name. Results depend on the current view screen. See <a href="#">"Search Actions" on page 62</a> for more information.</p> <p>These are the valid values:</p> <ol style="list-style-type: none"><li>1. Find Searches for one or all occurrences of the paragraph name. This is the default.</li><li>2. Highlight Highlights source code lines or boxes containing the paragraph name. Lines or boxes already highlighted are not reset.</li><li>3. Scroll Positions the screen to the first line containing the paragraph name. Previously highlighted lines remain unchanged. Scroll is valid on Source View only.</li><li>4. Print Copies lines containing the paragraph name to the List file. The List file is processed on the Options - Log/List/Punch Definition pop-up. Print is valid on Source View only.</li><li>5. Punch Copies lines containing the paragraph name to the Punch file. The Punch file is processed on the Options - Log/List/Punch Definition pop-up. Punch is valid on Source View only.</li><li>6. Exclude Omits lines containing occurrences of the paragraph name from the display. In Perform Structure View, the cursor is positioned to the first box representing a logical program unit that contains an excluded source code line.</li></ol>

---

## Search - Pattern String Pop-up

Use the Search - Pattern String pop-up to find, highlight, scroll, print, punch, or exclude a pattern, depending on the value in the Action field and the current view method.

To display the Search - Pattern String pop-up, follow this step:

- ▶ Select Search ▶ String and press Enter. The Search - Pattern String pop-up, shown in [Figure 40](#), displays.

**Figure 40 • Search - Pattern String Pop-up**

```

Search - Pattern String
Type a string and select search options. Then press Enter.
String _____

String type          Location
1  1. Simple         1  1. Any
   2. Hexadecimal    2  2. Word
   3. Text           3  3. Prefix
   4. Picture        4  4. Suffix

Direction           Options                Action
1  1. All            - IN-clause...        1  1. Find
   2. Next                               2  2. Highlight
   3. Previous                               3  3. Scroll
   4. First                               4  4. Print
   5. Last                               5  5. Punch
                                           6  6. Exclude

Begin column ____ End column ____

```

### Fields

Field	Description
String	Required. Specifies a string of alphanumeric or DBCS characters. It is not necessary to enclose the string in single or double quotes. All characters are used for the pattern search.
String type	Specifies the type of pattern string. These are the valid values: <ul style="list-style-type: none"> <li>1. Simple      Indicates a string that is not a hexadecimal, text, or a picture string. This is the default.</li> <li>2. Hexadecimal      Specific unprintable characters may be specified by giving the EBCDIC hexadecimal value.</li> <li>3. Text      A character string may be entered regardless of upper or lowercase by using the text option.</li> </ul>

Field	Description
4. Picture	<p>You may enter a string profile instead of exact characters. Insight defines nine special characters for use in picture strings. You can combine these special characters with other characters. These are the valid special characters:</p> <p>P'=' Any character</p> <p>P'-' Any non blank character</p> <p>P'.' Any non display character</p> <p>P'#" Any numeric character</p> <p>P'-' Any character</p> <p>P'@" Any character</p> <p>P'&lt;' Any character</p> <p>P'&gt;' Any character</p> <p>P'\$' Any character</p>
Location	<p>Specifies the location to search for the pattern string. These are the valid values:</p> <ol style="list-style-type: none"> <li>1. Any Specifies a string anywhere in the source code.</li> <li>2. Word Specifies a string preceded and followed by any non-alphanumeric character (except hyphen).</li> <li>3. Prefix Specifies a word that begins with the specified string.</li> <li>4. Suffix Specifies a word that ends with the specified string.</li> </ol>
Direction	<p>Restricts the search to a specific direction or occurrence. These are the valid values:</p> <ol style="list-style-type: none"> <li>1. All Searches for all occurrences of the string and displays the number found in the short/long message field. This is the default.</li> <li>2. Next Searches forward from a specified cursor position to the next occurrence of the string. Next is valid on Source View only.</li> <li>3. Previous Searches backward from a specified cursor position to the previous occurrence of the string. Previous is valid on Source View only.</li> </ol>

Field	Description
4. First	Searches from the top of the source file to the first occurrence of the string. First is valid on Source View only.
5. Last	Searches backward from the bottom of the source file to the last occurrence of the string. Last is valid on Source View only.
IN-clause	Displays the IN-Clause Option pop-up used to restrict the source lines or boxes to be considered for a search. If left blank, the entire source code program is searched. The default is the entire program. See <a href="#">"IN-Clause Option Pop-up" on page 97</a> for more information.
Action	<p>Specifies the action to be performed on the string. These are the valid values:</p> <p><b>Note:</b> _____ Results depend on the current view screen. See <a href="#">"Search Actions" on page 62</a> for more information.</p> <hr/> <p>1. Find      Searches for one or all occurrences of the string. This is the default.</p> <p>2. Highlight   Highlights source code lines or boxes containing the string. Lines or boxes that are already highlighted are not reset.</p> <p>3. Scroll      Positions the screen to the first line containing the string. Previously highlighted lines remain unchanged. Scroll is valid on Source View only.</p> <p>4. Print      Copies lines containing the string to the List file. The List file is processed on the Options - Log/List/Punch Definition pop-up. Print is valid on Source View only.</p> <p>5. Punch      Copies lines containing the string to the Punch file. The Punch file is processed on the Options - Log/List/Punch Definition pop-up. Punch is valid on Source View only.</p> <p>6. Exclude    Omits lines containing occurrences of the string from the display. In Perform Structure View, the cursor is positioned to the first box representing a logical program unit that contains an excluded source code line.</p>

Field	Description
Begin column	Specifies the column number where the search is to begin. The default is column 7.
End column	Specifies the column number where the search is to end. The default is column 66.

## Search - COBOL Subset Name Pop-up

Use the Search - COBOL Subset Name pop-up to find, highlight, scroll, print, punch, or exclude statements in COBOL subsets, depending on the value in the Action field and the current view method.

To display the Search - COBOL Subset Name pop-up, follow this step:

- ▶ Select Search ▶ Subset and press Enter. The Search - COBOL Subset Name pop-up, shown in [Figure 41](#), displays.

**Figure 41 • Search - COBOL Subset Name Pop-up**

```

Search - COBOL Subset Name

Type a subset name and select search options. Then press Enter.
For a selection list, type a pattern (e.g. ABC*) in the name area.

Subset name _____

Direction      Options      Action
1  1. All        - IN-clause...  1  1. Find
   2. Next                               2. Highlight
   3. Previous                               3. Scroll
   4. First                                4. Print
   5. Last                                  5. Punch
                                           6. Exclude
  
```

## Fields

Field	Description																																							
Subset name	<p>Specifies the name of a predefined COBOL subset. Subsets can be concatenated by placing a plus sign (+) between the subset names.</p> <p>These are the predefined subsets:</p> <table border="0"> <tr> <td>ASsignment</td> <td>CALL</td> <td>CIcs</td> </tr> <tr> <td>COBOLII</td> <td>COBOL/370</td> <td>COMment</td> </tr> <tr> <td>CONditional</td> <td>COPY</td> <td>DB2/SQL</td> </tr> <tr> <td>DDL</td> <td>DEAD</td> <td>DEADCode</td> </tr> <tr> <td>DEADData</td> <td>DEBUg</td> <td>DEFinition</td> </tr> <tr> <td>DIRective</td> <td>DIVision</td> <td>DL/I   DL/1</td> </tr> <tr> <td>DML</td> <td>ENtry</td> <td>EXIt   PGMExit</td> </tr> <tr> <td>FALLthrough</td> <td>GOto</td> <td>IDMS</td> </tr> <tr> <td>INCLude</td> <td>Input</td> <td>IO</td> </tr> <tr> <td>LABEL</td> <td>MAINline</td> <td>MATH</td> </tr> <tr> <td>Output</td> <td>PARagraph</td> <td>PERform</td> </tr> <tr> <td>RETurn</td> <td>SECTion</td> <td>SORTMerge</td> </tr> <tr> <td>STRucture</td> <td>TESTed</td> <td>UNTested</td> </tr> </table>	ASsignment	CALL	CIcs	COBOLII	COBOL/370	COMment	CONditional	COPY	DB2/SQL	DDL	DEAD	DEADCode	DEADData	DEBUg	DEFinition	DIRective	DIVision	DL/I   DL/1	DML	ENtry	EXIt   PGMExit	FALLthrough	GOto	IDMS	INCLude	Input	IO	LABEL	MAINline	MATH	Output	PARagraph	PERform	RETurn	SECTion	SORTMerge	STRucture	TESTed	UNTested
ASsignment	CALL	CIcs																																						
COBOLII	COBOL/370	COMment																																						
CONditional	COPY	DB2/SQL																																						
DDL	DEAD	DEADCode																																						
DEADData	DEBUg	DEFinition																																						
DIRective	DIVision	DL/I   DL/1																																						
DML	ENtry	EXIt   PGMExit																																						
FALLthrough	GOto	IDMS																																						
INCLude	Input	IO																																						
LABEL	MAINline	MATH																																						
Output	PARagraph	PERform																																						
RETurn	SECTion	SORTMerge																																						
STRucture	TESTed	UNTested																																						

**Note:**

See the *ASG-Insight User's Guide* for a description of each subset and subset type.

A screen subset:

Highlighted | HI

NONHighlighted | NHI

Excluded | X

NONExcluded | NX

Tagged lines subsets of tags in columns 73 through 80.

To display the Selection List pop-up for subsets, leave the Subset name field blank, or enter a pattern with wildcard characters [question mark (?) for one character; asterisk (\*) for zero or more characters].

See ["Selection List Pop-ups" on page 96](#) for more information.

Field	Description
Direction	<p>Restricts the search to a specific direction or occurrence. These are the valid values:</p> <ol style="list-style-type: none"> <li>1. All Searches for all occurrences of the subset and displays the number found in the short/long message field. This is the default.</li> <li>2. Next Searches forward from a specified cursor position to the next occurrence of the subset. Next is valid on Source View only.</li> <li>3. Previous Searches backward from a specified cursor position to the previous occurrence of the subset. Previous is valid on Source View only.</li> <li>4. First Searches from the top of the source file to the first occurrence of the subset. First is valid on Source View only.</li> <li>5. Last Searches backward from the bottom of the source file to the last occurrence of the subset. Last is valid on Source View only.</li> </ol>
IN-clause	<p>Displays the IN-Clause Option pop-up used to restrict the source lines or boxes to be considered for a search. If left blank, the entire source code program is searched. The default is the entire program.</p> <p>See <a href="#">"IN-Clause Option Pop-up" on page 97</a> for more information.</p>
Action	<p>Specifies the action to be performed on the subset. Results depend on the current view screen. See <a href="#">"Search Actions" on page 62</a> for more information. These are the valid actions:</p> <ol style="list-style-type: none"> <li>1. Find Searches for one or all occurrences of the subset. This is the default.</li> <li>2. Highlight Highlights source code lines or boxes containing the subset. Lines or boxes already highlighted are not reset.</li> <li>3. Scroll Positions the screen to the first line containing the subset. Previously highlighted lines remain unchanged. Scroll is valid on Source View only.</li> <li>4. Print Copies lines containing the subset to the List file. The List file is processed on the Options - Log/List/Punch Definition pop-up. Print is valid on Source View only.</li> </ol>

Field	Description
5. Punch	Copies lines containing the subset to the Punch file. The Punch file is processed on the Options - Log/List/Punch Definition pop-up.
6. Exclude	Omits lines containing occurrences of the subset from the display. In Perform Structure View, the cursor is positioned at the first box representing a logical program unit that contains an excluded source code line.

## Search - Perform Range Name Pop-up

Use the Search - Perform Range Name pop-up to find, highlight, scroll, print, punch, or exclude statements in a perform range, depending on the value in the Action field and the current view method.

To display the Search - Perform Range Name pop-up, follow this step:

- ▶ Select Search ▶ Perform and press Enter. The Search - Perform Range Name pop-up, shown in [Figure 42](#), displays.

**Figure 42 • Search - Perform Range Name Pop-up**

Search - Perform Range Name

Type a perfrange name and select search options. Then press Enter.  
For a selection list, type a pattern (e.g. ABC\*) in the name area.

Perfrange name \_\_\_\_\_

Direction	Options	Action
1 1. All	_ IN-clause...	1 1. Find
2. Next		2. Highlight
3. Previous		3. Scroll
4. First		4. Print
5. Last		5. Punch
		6. Exclude

## Fields

Field	Description
Perfrange Name	<p>Specifies the name in a PERFORM statement, or the name of any section contained in the DECLARATIVES.</p> <p>To display the Selection List pop-up, leave the Perfrange Name field blank, or enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters]. See <a href="#">"Selection List Pop-ups" on page 96</a> for more information.</p>
Direction	<p>Restricts the search to a specific direction or occurrence. These are the valid values:</p> <ol style="list-style-type: none"><li>1. All            Searches for all occurrences of the perform range and displays the number found in the short/long message field. This is the default.</li><li>2. Next           Searches forward from a specified cursor position to the next occurrence of the perform range. Next is valid on Source View only.</li><li>3. Previous       Searches backward from a specified cursor position to the previous occurrence of the perform range. Previous is valid on Source View only.</li><li>4. First           Searches from the top of the source file to the first occurrence of the perform range. First is valid on Source View only.</li><li>5. Last            Searches backward from the bottom of the source file to the last occurrence of the perform range. Last is valid on Source View only.</li></ol>
IN-clause	<p>Displays the IN-Clause Option pop-up used to restrict the source lines or boxes to be considered for a search. If left blank, the entire source code program is searched. The default is the entire program. See <a href="#">"IN-Clause Option Pop-up" on page 97</a> for more information.</p>

Field	Description
Action	<p>Specifies the action to be performed on the perform range. Results depend on the current view screen. See <a href="#">"Search Actions" on page 62</a> for more information. These are the valid actions:</p>
1. Find	Searches for one or all occurrences of the perform range. This is the default.
2. Highlight	Highlights source code lines or boxes containing the perform range. Lines or boxes already highlighted are not reset.
3. Scroll	Positions the screen to the first line containing the perform range. Previously highlighted lines remain unchanged. Scroll is valid on Source View only.
4. Print	Copies lines containing the perform range to the List file. The List file is processed on the Options - Log/List/Punch Definition pop-up. Print is valid on Source View only.
5. Punch	Copies lines containing the perform range to the Punch file. The Punch file is processed on the Options - Log/List/Punch Definition pop-up. Punch is valid on Source View only.
6. Exclude	Omits lines containing occurrences of the perform range from the display. In Perform Structure View, the cursor is positioned to the first box representing a logical program unit that contains an excluded source code line.

## Search - Program Name Pop-up

Use the Search - Program Name pop-up to find, highlight, scroll, print, punch, or exclude statements in sub-programs, depending on the value in the Action field and the current view method.

To display the Search - Program Name pop-up, follow this step:

- ▶ Select Search ▶ Program and press Enter. The Search - Program Name pop-up, shown in [Figure 43](#), displays.

**Figure 43 • Search - Program Name Pop-up**

Search - Program Name

Type a (sub)program name and select search options. Then press Enter. For a selection list, type a pattern (e.g. ABC\*) in the name area.

Program name \_\_\_\_\_

Direction	Options	Action
1 1. All	_ IN-clause...	1 1. Find
2. Next		2. Highlight
3. Previous		3. Scroll
4. First		4. Print
5. Last		5. Punch
		6. Exclude

### Fields

Field	Description				
Program name	<p>Specifies the name of the main program or any nested program.</p> <p>To display the Selection List pop-up, leave the Program name field blank, or enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters]. See <a href="#">"Selection List Pop-ups" on page 96</a> for more information.</p> <p>Program names may be concatenated by placing a plus sign (+) between the program names.</p>				
Direction	<p>Restricts the search to a specific direction or occurrence. These are the valid values:</p> <table style="width: 100%; border-collapse: collapse;"> <tbody> <tr> <td style="width: 20%;">1. All</td> <td>Searches for all occurrences of the program name and displays the number found in the short/long message field. This is the default.</td> </tr> <tr> <td>2. Next</td> <td>Searches forward from a specified cursor position to the next occurrence of the program name. Next is valid on Source View only.</td> </tr> </tbody> </table>	1. All	Searches for all occurrences of the program name and displays the number found in the short/long message field. This is the default.	2. Next	Searches forward from a specified cursor position to the next occurrence of the program name. Next is valid on Source View only.
1. All	Searches for all occurrences of the program name and displays the number found in the short/long message field. This is the default.				
2. Next	Searches forward from a specified cursor position to the next occurrence of the program name. Next is valid on Source View only.				

Field	Description
	<p>3. Previous Searches backward from a specified cursor position to the previous occurrence of the program name. Previous is valid on Source View only.</p> <p>4. First Searches from the top of the source file to the first occurrence of the program name. First is valid on Source View only.</p> <p>5. Last Searches backward from the bottom of the source file to the last occurrence of the program name. Last is valid on Source View only.</p>
IN-clause	<p>Displays the IN-Clause Option pop-up used to restrict the source lines or boxes to be considered for a search. If left blank, the entire source code program is searched.</p> <p>The default is the entire program. See <a href="#">"IN-Clause Option Pop-up" on page 97</a> for more information.</p>
Action	<p>Specifies the action to be performed on the program name. Results depend on the current view screen. See <a href="#">"Search Actions" on page 62</a> for more information. These are the valid actions:</p> <p>1. Find Searches for one or all occurrences of the program name. This is the default.</p> <p>2. Highlight Highlights source code lines or boxes containing the program name. Lines or boxes already highlighted are not reset.</p> <p>3. Scroll Positions the screen to the first line containing the program name. Previously highlighted lines remain unchanged. Scroll is valid on Source View only.</p> <p>4. Print Copies lines containing the specified target to the List file. The List file the Options - Log/List/Punch Definition pop-up. Print is valid on Source View only.</p>

Field	Description
5. Punch	Copies lines containing the specified target to the Punch file. The Punch file is processed on the Options - Log/List/Punch Definition pop-up. Punch is valid on Source View only.
6. Exclude	Omits lines containing occurrences of the program name from the display. In Perform Structure View, the cursor is positioned to the first box representing a logical program unit that contains an excluded source code line.

## Search - User Mark Name Pop-up

Use the Search - User Mark Name pop-up to find, highlight, scroll, print, punch, or exclude sets of marked lines, depending on the value in the Action field and the current view method.

To display the Search - User Mark Name pop-up, follow this step:

- ▶ Select Search ▶ Mark and press Enter. The Search - User Mark Name pop-up, shown in [Figure 44](#), displays.

**Figure 44 • Search - User Mark Name Pop-up**

Search - User Mark Name

Type a mark name and select search options. Then press Enter. For a selection list, type a pattern (e.g. ABC\*) in the name area.

Mark name \_\_\_\_\_

Direction	Options	Action
1 1. All	_ IN-clause...	1 1. Find
2. Next		2. Highlight
3. Previous		3. Scroll
4. First		4. Print
5. Last		5. Punch
		6. Exclude

## Fields

Field	Description
Mark name	<p>Specifies a 1 to 10 alphanumeric character name given to a set or path.</p> <p>To display the Selection List pop-up, leave the Mark Name field blank or enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters].</p> <p>See <a href="#">"Selection List Pop-ups" on page 96</a> for more information.</p>
Direction	<p>Restricts the search to a specific direction or occurrence. These are the valid values:</p> <ol style="list-style-type: none"> <li>1. All Searches for all occurrences of the mark name and displays the number found in the short/long message field. This is the default.</li> <li>2. Next Searches forward from a specified cursor position to the next occurrence of the mark name. Next is valid on Source View only.</li> <li>3. Previous Searches backward from a specified cursor position to the previous occurrence of the mark name. Previous is valid on Source View only.</li> <li>4. First Searches from the top of the source file to the first occurrence of the mark name. First is valid on Source View only.</li> <li>5. Last Searches backward from the bottom of the source file to the last occurrence of the mark name. Last is valid on Source View only.</li> </ol>
IN-clause	<p>Displays the IN-Clause Option pop-up used to restrict the source lines or boxes to be considered for a search. If left blank, the entire source code program is searched. The default is the entire program. See <a href="#">"IN-Clause Option Pop-up" on page 97</a> for more information.</p>
Action	<p>Specifies the action to be performed on the mark name. Results depend on the current view screen. See <a href="#">"Search Actions" on page 62</a> for more information. These are the valid actions:</p> <ol style="list-style-type: none"> <li>1. Find Searches for one or all occurrences of the mark name. This is the default.</li> <li>2. Highlight Highlights source code lines or boxes containing the mark name. Lines or boxes already highlighted are not reset.</li> </ol>

Field	Description
3. Scroll	Positions the screen to the first line containing the mark name. Previously highlighted lines remain unchanged. Scroll is valid on Source View only.
4. Print	Copies lines containing the mark name to the List file. The List file is processed on the Options - Log/List/Punch Definition pop-up. Print is valid on Source View only.
5. Punch	Copies lines containing the mark name to the Punch file. The Punch file is processed on the Options - Log/List/Punch Definition pop-up. Punch is valid on Source View only.
6. Exclude	Omits lines containing occurrences of the mark name from the display. In Perform Structure View, the cursor is positioned to the first box representing a logical program unit that contains an excluded source code line.

## Search - Line Range Pop-up

Use the Search - Line Range pop-up to find, highlight, scroll, print, punch, or exclude a single line or range of lines, depending on the value in the Action field and the current view method.

To display the Search - Line Range pop-up, follow this step:

- ▶ Select Search ▶ Line and press Enter. The Search - Line Range pop-up, shown in [Figure 45](#), displays.

**Figure 45 • Search - Line Range Pop-up**

Search - Line Range

Type a line range (or line) and select search options. Then press Enter.

Line range \_\_\_\_\_

Direction	Options	Action
1 1. All	_ IN-clause...	1 1. Find
2. Next		2. Highlight
3. Previous		3. Scroll
4. First		4. Print
5. Last		5. Punch
		6. Exclude

## Fields

Field	Description
Line Range	Specifies a single line number or range of lines. Line numbers display in columns 1 through 6 of the Source View.
Direction	<p>Restricts the search to a specific direction or occurrence. These are the valid values:</p> <ol style="list-style-type: none"> <li>1. All Searches for all occurrences of the line range and displays the number found in the short/long message field. This is the default.</li> <li>2. Next Searches forward from a specified cursor position to the next occurrence of the line range. Next is valid on Source View only.</li> <li>3. Previous Searches backward from a specified cursor position to the previous occurrence of the line range. Previous is valid on Source View only.</li> <li>4. First Searches from the top of the source file to the first occurrence of the line range. First is valid on Source View only.</li> <li>5. Last Searches backward from the bottom of the source file to the last occurrence of the line range. Last is valid on Source View only.</li> </ol>
IN-clause	Displays the IN-Clause Option pop-up used to restrict the source lines or boxes to be considered for a search. If left blank, the entire source code program is searched. The default is the entire program. See <a href="#">"IN-Clause Option Pop-up" on page 97</a> for more information.
Action	<p>Specifies the action to be performed on the line range. These are the valid actions:</p> <ol style="list-style-type: none"> <li>1. Find Searches for one or all occurrences of the line range. This is the default.</li> <li>2. Highlight Highlights source code lines or boxes containing the line range. Lines or boxes already highlighted are not reset.</li> </ol>

Field	Description
3. Scroll	Positions the screen to the first line containing the line range. Previously highlighted lines remain unchanged. Scroll is valid on Source View only.
4. Print	Copies lines containing the line range to the List file. The List file is processed on the Options - Log/List/Punch Definition pop-up. Print is valid on Source View only.
5. Punch	Copies lines containing the line range to the Punch file. The Punch file is processed on the Options - Log/List/Punch Definition pop-up. Punch is valid on Source View only.
6. Exclude	Omits lines containing occurrences of the line range from the display. In Perform Structure View, the cursor is positioned to the first box representing a logical program unit that contains an excluded source code line.

Results depend on the current view screen. See ["Search Actions" on page 62](#) for more information.

---

## Search - Any/Unknown Type Pop-up

Use the Search - Any/Unknown Type pop-up to find, highlight, scroll, print, punch, or exclude targets when the type is unknown, does not fit one of the predefined categories, or is a combination of predefined categories.

To display the Search - Any/Unknown Type pop-up, follow this step:

- ▶ Select Search ▶ Any and press Enter. The Search - Any/Unknown Type pop-up, shown in [Figure 46](#), displays.

**Figure 46 • Search - Any/Unknown Type Pop-up**

Search - Any/Unknown Type

To find references to a target of "unknown" type (data, label, etc.), type the target and select search options. Then press Enter

Target \_\_\_\_\_

Direction	Options	Action
1	1. All	1
	2. Next	2. Highlight
	3. Previous	3. Scroll
	4. First	4. Print
	5. Last	5. Punch
		6. Exclude

### Fields

Field	Description						
Target	<p>Specifies the name of the target. Target names can be concatenated by placing a plus sign (+) between them. To reuse the target of the previous search, type an asterisk (*) in the Target field.</p> <p>For COBOL II Release 3 or later programs, a dataname that may be ambiguous or used multiple times can be qualified by using OF followed by the program name.</p>						
Direction	<p>Indicates the direction for the search. These are the valid values:</p> <table style="width: 100%; border-collapse: collapse;"> <tbody> <tr> <td style="width: 30%;">1. All</td> <td>Searches for all occurrences of the target and displays the number found in the short/long message field. This is the default.</td> </tr> <tr> <td>2. Next</td> <td>Searches forward from a specified cursor position to the next occurrence of the target.</td> </tr> <tr> <td>3. Previous</td> <td>Searches backward from a specified cursor position to the previous occurrence of the target.</td> </tr> </tbody> </table>	1. All	Searches for all occurrences of the target and displays the number found in the short/long message field. This is the default.	2. Next	Searches forward from a specified cursor position to the next occurrence of the target.	3. Previous	Searches backward from a specified cursor position to the previous occurrence of the target.
1. All	Searches for all occurrences of the target and displays the number found in the short/long message field. This is the default.						
2. Next	Searches forward from a specified cursor position to the next occurrence of the target.						
3. Previous	Searches backward from a specified cursor position to the previous occurrence of the target.						

Field	Description
	<ol style="list-style-type: none"><li>4. First Searches from the top of the source file to the first occurrence of the target.</li><li>5. Last Searches backward from the bottom of the source file to the last occurrence of the target.</li></ol>
IN-clause	Displays the IN-Clause Option pop-up used to restrict the source lines or boxes to be considered for a search. If left blank, the entire source code program is searched. The default is the entire program. See <a href="#">"IN-Clause Option Pop-up" on page 97</a> for more information.
Action	Specifies the action to be performed on the target. Results depend on the current view screen. See <a href="#">"Search Actions" on page 62</a> for more information. These are the valid actions: <ol style="list-style-type: none"><li>1. Find Searches for one or all occurrences of the specified target.</li><li>2. Highlight Highlights source code lines or boxes containing the specified targets. Lines or boxes already highlighted are not reset.</li><li>3. Scroll Positions the screen to the first line or box containing the specified target. Previously highlighted lines or boxes remain unchanged.</li><li>4. Print Copies lines or boxes containing the specified target to the List file. The List file is processed on the Options - Log/List/Punch Definition pop-up.</li><li>5. Punch Copies lines or boxes containing the specified target to the Punch file for subsequent processing. The Punch file is processed on the Options - Log/List/Punch Definition pop-up.</li><li>6. Exclude Omits lines containing occurrences of the target from the display.</li></ol>

## Search - Branch Request Pop-up

To position the cursor at the target on the Source View screen, follow this step:

- ▶ Select Search ▶ Branch and press Enter. The Search - Branch Request pop-up, shown in [Figure 47](#), displays.

**Figure 47 • Search - Branch Request Pop-up**

Search - Branch Request

To branch to another area of the program, select the Option desired  
For Option 1, type the branch location (Target) information. Then  
press Enter. For a name selection list (for Target type 2, 3 or 4),  
type a pattern (e.g. ABC\*) in the name field.

Option

1 1. Branch to target  
2. Return to previous "Branch to target" location  
3. Branch to transfer(s) of control to cursor position

Target name \_\_\_\_\_

Target type

1 1. None - use cursor  
2. Label name  
3. Perfrange name  
4. Program name

### Fields

Field	Description
Option	Specifies the type of branch to perform. These are the valid options:
Branch to target	Positions the cursor to the name specified in the Target name field. If you select this Option, the Target name field is required. This is the default.
Return to branch location	Positions the cursor to the statement from where the previous branch occurred.
Display all transfers of control to cursor position	<p>Redisplays the Source View screen with a long message telling you to position the cursor to the desired branch location and press Enter. The cursor is positioned to the statement that branches to the specified branch location.</p> <p>If more than one statement transfers control to the specified branch location, the Branch Previous Options pop-up displays allowing you to select the desired statement.</p> <p>If this Option is selected, the Target name and Target type fields are invalid.</p>

Field	Description
Target name	<p>Required if Option 1 is selected. Type a Perform Range name, label name, paragraph name, or section name. For COBOL II Release 3 or later programs, you can also enter a subprogram name.</p> <p>For Perform Ranges and label names, if the specified name is not fully qualified and there are multiple Perform Ranges or labels with the specified name, the cursor is positioned on the first statement found.</p> <p>For COBOL II Release 3 or later programs, a data item, label or program name that may be ambiguous or used multiple times can be qualified by using OF followed by the program name.</p> <p>To display the Selection List pop-up, leave the Target name blank or enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters], and select Target type 2, 3, or 4. On the Selection List pop-up, select the desired name by typing S next to the desired name. See <a href="#">"Selection List Pop-ups" on page 96</a> for more information.</p>
Target type	<p>Specifies the number of the target type corresponding to the entry in the Target name field. This field is valid for Option 1 only. These are the valid target types:</p>
None - Use Cursor	<p>Redisplays the Source View screen with a long message stating to position the cursor to the selected statement and press Enter. This is the default.</p> <p>If the statement is a GO TO, PERFORM, or internal CALL, the cursor is positioned to the target of that statement. If you select any other statement type, the cursor is positioned to the next sequential statement to be executed. The PROCEDURE DIVISION label is located if the cursor is positioned outside the PROCEDURE DIVISION</p>
Label Name	<p>Positions the cursor to the specified paragraph or section name. If the specified name is not fully qualified and there are multiple paragraphs or sections with the specified name, the cursor is positioned to the first paragraph or section found.</p>
Perfrange Name	<p>Positions the cursor to the specified Perform Range. If the specified name is not fully qualified and there are multiple Perform Ranges with the specified name, the cursor is positioned to the first Perform Range found.</p>
Subprogram Name	<p>Positions the cursor to the specified nested program. For COBOL II Release 3 and later programs, a data item, label, or program name that may be ambiguous or used multiple times can be qualified by using OF followed by the program name.</p>

## Branch Previous Options Pop-up

To select a statement for branching, follow this step:

- ▶ Type `BRANCH PREVIOUS`. This pop-up displays when more than one statement transfers control to the specified branch location on the Search - Branch Request pop-up. The Branch Previous Options pop-up, shown in [Figure 48](#), displays.

**Figure 48 • Branch Previous Options Pop-up**

```

Branch Previous Options                                LINE 1 OF 3
Command ==> _____ Scroll ==> CSR

Select the statement to be viewed.  Then press Enter.

S  Line      Source statement.
-----
- 000300      PERFORM P999-ABEND-PROGRAM.
- 000302      PERFORM P999-ABEND-PROGRAM.
- 000448      PERFORM P999-ABEND-PROGRAM
*****
***** BOTTOM OF DATA *****

```

When this pop-up first displays, the short message field indicates the number of statements found.

### Fields

Field	Description
Line command area	The area to the left of the Line field, which accepts the S line command. Type S to select the identified source statement and to position the cursor to that statement on the Source View screen.
Line	Specifies the source statement line number.
Source statement	Specifies the source code statement.

## Selection List Pop-ups

### *To display the Selection List pop-up*

- 1 Leave the name field blank.

Or

Enter a pattern with wildcard characters [question mark (?) for one character; asterisk (\*) for zero or more characters] in the name field. The Selection List pop-up, shown in [Figure 49](#), displays. (The selection list pop-ups are the same except for the type of list displayed.)

Selection list pop-ups are available for the data, perform range, paragraph, program, mark, subset, and branch searches.

**Figure 49 • Selection List - Data Names Pop-up**

```

Command ==> _____ Selection List - Data Names _____ Scroll ==> CSR
S   Data name
- -----
- ABEND-CODE
- ADDRESS1
- AREA-CODE
- AVG-AMT (DEAD)
- BILLING-DAYS
- CHECK-CODE (DEAD)
- CITY
- CLIENT-ID
- CONSOLE
- CUR-PREFIX
- CURRENT-ZIP-DATA
- CUSTOMER-ID
- DBA-DEPT-CODE
- DEBUG-PARM
- DET-ADDRESS
- DET-AREA-CODE
- DET-BILLING-DAYS
-

```

- 2 Select a name from the list by typing S in the line command area. You can select more than one name on these pop-ups:
  - Selection List - Data Names
  - Selection List - Label/Paragraph Names
  - Selection List - Subset Names
  - Selection List - Program Names
- 3 Press Enter to redisplay the Search pop-up with the selected name(s) in the appropriate field.

## IN-Clause Option Pop-up

To restrict the source lines considered for a search, follow this step:

- Specify the IN-clause option on a Search pop-up. The IN-Clause Option pop-up, shown in [Figure 50](#), displays.

**Figure 50 • IN-Clause Option Pop-up**

IN-Clause Option

To restrict the source lines to be considered, type one or more IN-Clause names. Then press Enter. For a selection list, type a pattern in the appropriate name field.

Line range . . . \_\_\_\_\_  
 Subset name . . . \_\_\_\_\_  
 Paragraph Name . \_\_\_\_\_  
 Perfrange name . \_\_\_\_\_  
 Program name . . \_\_\_\_\_  
 User MARK name . \_\_\_\_\_

---

4. First	4. Print
5. Last	5. Punch
	6. Exclude

## Fields

Field	Description																																							
Line range	Specifies a single line number or range of lines.																																							
Subset name	Specifies a predefined COBOL language subset name. These are the valid subsets: <table style="margin-left: 20px; border: none;"> <tr> <td>ASsignment</td> <td>CALL</td> <td>CIcs</td> </tr> <tr> <td>COBOLII</td> <td>COBOL/370</td> <td>COMment</td> </tr> <tr> <td>CONditional</td> <td>COPy</td> <td>DB2/SQL</td> </tr> <tr> <td>DDL</td> <td>DEAD</td> <td>DEADCode</td> </tr> <tr> <td>DEADData</td> <td>DEBug</td> <td>DEFinition</td> </tr> <tr> <td>DIRective</td> <td>DIVision</td> <td>DL/I   DL/1</td> </tr> <tr> <td>DML</td> <td>ENtry</td> <td>EXIt   PGMExit</td> </tr> <tr> <td>FALLthrough</td> <td>GOto</td> <td>IDMS</td> </tr> <tr> <td>INclude</td> <td>Input</td> <td>IO</td> </tr> <tr> <td>LABEL</td> <td>MAINline</td> <td>MATH</td> </tr> <tr> <td>Output</td> <td>PARagraph</td> <td>PERform</td> </tr> <tr> <td>RETurn</td> <td>SECTion</td> <td>SORTMerge</td> </tr> <tr> <td>STructure</td> <td>TESTed</td> <td>UNTested</td> </tr> </table>	ASsignment	CALL	CIcs	COBOLII	COBOL/370	COMment	CONditional	COPy	DB2/SQL	DDL	DEAD	DEADCode	DEADData	DEBug	DEFinition	DIRective	DIVision	DL/I   DL/1	DML	ENtry	EXIt   PGMExit	FALLthrough	GOto	IDMS	INclude	Input	IO	LABEL	MAINline	MATH	Output	PARagraph	PERform	RETurn	SECTion	SORTMerge	STructure	TESTed	UNTested
ASsignment	CALL	CIcs																																						
COBOLII	COBOL/370	COMment																																						
CONditional	COPy	DB2/SQL																																						
DDL	DEAD	DEADCode																																						
DEADData	DEBug	DEFinition																																						
DIRective	DIVision	DL/I   DL/1																																						
DML	ENtry	EXIt   PGMExit																																						
FALLthrough	GOto	IDMS																																						
INclude	Input	IO																																						
LABEL	MAINline	MATH																																						
Output	PARagraph	PERform																																						
RETurn	SECTion	SORTMerge																																						
STructure	TESTed	UNTested																																						

Field	Description
	<p><b>Note:</b></p> <p>See the <i>ASG-Insight User's Guide</i> for a description of each subset and subset type</p> <p>A screen subset:</p> <p>Highlighted   HI</p> <p>NONHighlighted   NHI</p> <p>Excluded   X</p> <p>NONExcluded   NX</p> <p>Tagged lines subsets of tags in columns 73 through 80.</p>
Paragraph Name	Specifies a PROCEDURE DIVISION paragraph or section name. The PROCEDURE and PROC literals can also be specified.
Perfrange name	Optional. Type the name specified in a PERFORM statement or the name of any section contained in the DECLARATIVES. The search includes all statements executed as a result of the PERFORM statement.
Program name	Optional. Type the name of the main program or any nested program. The search includes all code contained in the program, including all programs physically nested inside the specified program.

## Usage Notes

In most cases, you may specify any combination of IN-clause options. However, some Insight-defined mark names are incompatible with other logical sets of COBOL statements and the combinations are not allowed.

All fields on this pop-up are optional. If all fields are left blank, the search is not restricted (i.e., the entire program is searched).

---

# 5

## Logic

---

This chapter describes items on the Insight Logic menu and contains these sections:

Topic	Page
<a href="#">Logic Pull-down</a>	<a href="#">101</a>
<a href="#">Logic Order Search - Data Name Pop-up</a>	<a href="#">102</a>
<a href="#">Logic Order Search - Label Name Pop-up</a>	<a href="#">105</a>
<a href="#">Logic Order Search - COBOL Subset Name Pop-up</a>	<a href="#">107</a>
<a href="#">Logic Order Search - Perform Range Name Pop-up</a>	<a href="#">110</a>
<a href="#">Logic Order Search - User Mark Name Pop-up</a>	<a href="#">112</a>
<a href="#">Logic Order Search - Subprogram Name Pop-up</a>	<a href="#">114</a>
<a href="#">Logic Order Search - Line Number Range Pop-up</a>	<a href="#">116</a>
<a href="#">Selection List Pop-ups</a>	<a href="#">118</a>
<a href="#">Logic Order Search - Trace IN-Clause Option Pop-up</a>	<a href="#">119</a>

The Logic facility is a convenient way to view paths in logical execution order, and to see how and where specific information is used. The logic order search displays results in these three ways:

- Show references only- highlights the targets on the path from the starting point to where the logic order search ends. For example, if the target is Data name and Uses is selected for References with Forward as the Direction, the highlighted result would be all the COBOL source statements where a USE of this dataname exists and are reachable in execution order from the starting point.

- Show execution path(s) from starting point to reference(s) - highlights all executable statements on the path from the starting point to where the logic order search ends. This path of executable statements from the starting point to each of the resulting targets is retained in a path named NETWORK. NETWORK has all the paths that take you from the starting point of the logic order search to all the results. The parts of NETWORK that go to the individual results are labeled SUBNET1 through SUBNET $n$ , where  $n$  is the number of results that were found.

A new NETWORK is created each time you use this Action. Use the Scratchpad pop-ups or the MARK command to copy and save a NETWORK.

You can use saved paths for additional analysis when they are used as targets for other Source View searches or commands.

- Trace (single-step) - stops at each conditional statement on the path from the starting point to the target of the search. A message indicates the reason for stopping. For a decision point, the line number where the trace function stopped and the line number to with continue are shown.

All executable statements in the path between a decision and a new decision point are highlighted. This path of executable statements is retained in a path named TRACK. A new TRACK is created each time this Action is used. The Scratchpad pop-ups or MARK command can be used to copy and save a TRACK.

This table contains information tags displayed in columns 73 through 80 for each highlighted line:

Tag	Description
DECISION	Specifies the decision point.
STOP PNT	Specifies the intermediate (informational) stopping point.
START	Specifies the beginning of the trace function.
OPTION $n$	Specifies the option choice.
TARGET	Specifies the destination of the trace function.

Multiple information tags on a line are separated with a forward slash (/).



Action	Description
6. Program	Displays the Logic Order Search - Subprogram Name pop-up used to follow program logic in subprograms.
7. Line	Displays the Logic Order Search - Line Number Range pop-up used to follow program logic within ranges of lines.

## Logic Order Search - Data Name Pop-up

To follow program logic to a specific dataname, follow this step:

- ▶ Select Logic ▶ Data and press Enter. The Logic Order Search - Data pop-up, shown in [Figure 52](#), displays.

**Figure 52 • Logic Order Search - Data Name Pop-up**

```

Logic Order Search - Data Name

To find data name references in execution order, type the name and
starting point, and select search options. Then press Enter. For a
data or label name selection list, type a pattern (e.g. ABC*) in the
appropriate input area.

Data name _____
Start line/label (if any) _____

References          Direction          Options
1  1. All           1  1. Next           - Trace
   2. Uses only    2  2. Previous      - IN-clause...
   3. Mods only    3  3. Forward (all)
                   4  4. Backward (all)

Action
1  1. Show data name reference(s)
   2. Show execution path(s) from starting point to reference(s)
   3. Trace (single-step) from starting point to reference(s)

```

## Fields

Field	Description
Data name	<p>Specifies a COBOL dataname or qualified COBOL dataname. Dataname refers to any valid COBOL reference for a data element.</p> <p>To display the Selection List pop-up, leave the Data name field blank or enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters].</p> <p>See <a href="#">"Selection List Pop-ups" on page 118</a> for more information.</p>
Start line/label	<p>Specifies the line number or label name from where to begin the search.</p> <p>To use cursor position for the beginning of the search, leave this field blank. A message prompting you to verify cursor position displays.</p> <p>To display the Selection List pop-up, enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters].</p> <p>See <a href="#">"Selection List Pop-ups" on page 118</a> for more information.</p>
References	<p>Restricts the trace to only those datanames that are defined, used, or modified, by specifying the type of dataname references. These are the valid values:</p> <ol style="list-style-type: none"> <li>1. All Includes occurrences of the dataname where its value is defined, tested, used, set, or modified. This is the default.</li> <li>2. Uses only Includes occurrences of the dataname where its value is being tested or used.</li> <li>3. Mods only Includes occurrences of the dataname where its value is being set or modified.</li> </ol>
Direction	<p>Optional. The logical search can be restricted to a specific direction or occurrence by specifying a direction. These are the valid values:</p> <ol style="list-style-type: none"> <li>1. Next Searches or traces forward from the starting point to the next occurrence of the target in each execution path. This is the default.</li> <li>2. Previous Searches or traces backward from the starting point to the previous occurrence of the target in each execution path.</li> <li>3. Forward Searches or traces forward from the starting point to all occurrences of the target.</li> <li>4. Backward Searches or traces backward from the starting point to all occurrences of the target.</li> </ol>

Direction	Description
Trace IN-Clause	<p>Displays the Logic Order Search - Trace IN-Clause Option pop-up to narrow the scope of the logic order trace to a specified perform range or marked set of lines. If not specified, the scope includes the entire program. You can only use this option if you selected number 3 for Action. The default is the entire program is used for the trace.</p> <p>See <a href="#">"Logic Order Search - Trace IN-Clause Option Pop-up" on page 119</a> for more information.</p>
Action	<p>Specifies how to display the results logic order search results. These are the valid actions:</p> <ol style="list-style-type: none"><li>1. Show Data Name Reference(s) Highlights only the datanames on the path from the starting point to where the logic order search ends. This is the default.</li><li>2. Show Execution Path(s) from Starting Point to Reference(s) All executable statements on the path from the starting point to where the logic order search stops are highlighted.</li><li>3. Trace (single-step) from Starting Point to Reference(s) The logic order search stops at all conditional statements on the path from the starting point to the target of the search.</li></ol>

## Logic Order Search - Label Name Pop-up

To follow program logic to a specific label name, follow this step:

- ▶ Select Logic ▶ Label and press Enter. The Logic Order Search - Label Name pop-up, shown on [Figure 53 on page 105](#), displays.

**Figure 53 • Logic Order Search - Label Name Pop-up**

```

Logic Order Search - Label Name

To find label references in execution order, type the label and
starting point, and select search options. Then press Enter. For a
label name selection list, type a pattern (e.g. ABC*) in the
appropriate input area.

Label name _____

Start line/label (if any) _____

Direction          Options
1  1. Next          _ Trace IN-clause...
   2. Previous
   3. Forward (all)
   4. Backward (all)

Action
1  1. Show label name reference(s)
   2. Show execution path(s) from starting point to reference(s)
   3. Trace (single-step) from starting point to reference(s)

```

### Fields

Field	Description
Label name	<p>Specifies any PROCEDURE DIVISION paragraph name or section name. The PROCEDURE and PROC literals can also be specified.</p> <p>To display the Selection List pop-up, leave the Label name field blank or enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters].</p> <p>See <a href="#">"Selection List Pop-ups" on page 118</a> for more information.</p>
Start line/label	<p>Specifies the line number or label name from where to begin the trace.</p> <p>To use cursor position for the beginning of the search, leave this field blank. A message prompting you to verify cursor position displays.</p> <p>To display the Selection List pop-up, enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters].</p> <p>See <a href="#">"Selection List Pop-ups" on page 118</a> for more information.</p>

Field	Description
Direction	<p>Restricts the logical search to a specific direction or occurrence. These are the valid values:</p> <ol style="list-style-type: none"><li>1. Next Searches or traces forward from the cursor position to the next occurrence of the target. This is the default.</li><li>2. Previous Searches or traces backward from the cursor position to the previous occurrence of the target.</li><li>3. Forward Searches or traces forward from the cursor position to all occurrences of the target.</li><li>4. Backward Searches or traces backward from the cursor position to all occurrences of the target.</li></ol>
Trace IN-clause	<p>Displays the Logic Order Search - Trace IN-Clause Option pop-up to narrow the scope of the logic order search to a specified perform range or marked set of lines. If not specified, the scope includes the entire program. This option can only be used if number 3 is selected for Action. The default is the entire program is used for the trace.</p> <p>See <a href="#">"Logic Order Search - Trace IN-Clause Option Pop-up" on page 119</a> for more information.</p>
Action	<p>Specifies how to display the results of the logic order search. These are the valid actions:</p> <ol style="list-style-type: none"><li>1. Show Label Name Reference(s) Only the targets on the path from the starting point to where the logic order search ends are highlighted. This is the default.</li><li>2. Show Execution Path(s) from Starting Point to Reference(s) All executable statements on the path from the starting point to where the logic order search stops are highlighted.</li><li>3. Trace (single-step) from Starting Point to Reference(s) The logic order search stops at all conditional statements on the path from the starting point to the target of the search.</li></ol>

## Logic Order Search - COBOL Subset Name Pop-up

To follow program logic to a COBOL subset, follow this step:

- ▶ Select Logic ▶ Subset and press Enter. The Logic Order Search - COBOL Subset Name pop-up, shown in [Figure 54](#), displays.

**Figure 54 • Logic Order Search - COBOL Subset Name Pop-up**

```

Logic Order Search - COBOL Subset Name

To find subset references in execution order, type the subset name
and starting point, and select search options. Then press Enter.
For a subset or label name selection list, type a pattern (e.g.
ABC*) in the appropriate input area.

Subset name _____
Start line/label (if any) _____

Direction          Options
1  1. Next          - Trace IN-clause...
   2. Previous
   3. Forward (all)
   4. Backward (all)

Action
1  1. Show subset name reference(s)
   2. Show execution path(s) from starting point to reference(s)
   3. Trace (single-step) from starting point to reference(s)

```

### Fields

Field	Description
Subset name	Specifies the name of a predefined COBOL subset. These are the valid subsets:
	ASsignment      CALL      Cics
	COBOLII      COBOL/370      COMment
	CONditional      COPy      DB2/SQL
	DDL      DEAD      DEADCode
	DEADData      DEBug      DEFinition
	DIRective      DIVision      DL/I   DL/1
	DML      ENtry      EXIt   PGMExit
	FALLthrough      GOTO      IDMS
	INclude      Input      IO
	LABEL      MAINline      MATH
	Output      PARagraph      PERform

Field	Description
	<p>RETurn            SEctIon            SORTMerge</p> <p>SStructure        TESTed            UNTested</p>
	<p><b>Note:</b></p> <p>See the <i>ASG-Insight User's Guide</i> for a description of each subset and subset type.</p> <p>These are the screen subsets:</p> <p>Highlighted   HI  NONHighlighted   NHI  Excluded   X  NONExcluded   NX</p> <p>Tagged lines subsets of tags in columns 73 through 80.</p> <p>To display the Selection List pop-up for subsets, leave the Subset name field blank, or enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters].</p> <p>See <a href="#">"Selection List Pop-ups" on page 118</a> for more information.</p>
Start line/label	<p>Specifies the line number or label name to begin the trace.</p> <p>To use cursor position for the beginning of the search, leave this field blank. A message prompting you to verify cursor position displays.</p> <p>To display the Selection List pop-up, enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters].</p> <p>See <a href="#">"Selection List Pop-ups" on page 118</a> for more information.</p>
Direction	<p>Restricts the logical search to a specific direction or occurrence by specifying a direction. These are the valid values:</p> <ol style="list-style-type: none"> <li>1. Next            Searches or traces forward from the cursor position to the next occurrence of the target. This is the default.</li> <li>2. Previous       Searches or traces backward from the cursor position to the previous occurrence of the target.</li> <li>3. Forward        Searches or traces forward from the cursor position to all occurrences of the target.</li> <li>4. Backward       Searches or traces backward from the cursor position to all occurrences of the target.</li> </ol>

---

Field	Description
Trace IN-clause	<p>Displays the Logic Order Search - Trace IN-Clause Option pop-up to narrow the scope of the logic order search to a specified perform range or marked set of lines. If not specified, the scope includes the entire program. This option can only be used if number 3 is selected for Action. The default is the entire program is used for the trace.</p> <p>See "<a href="#">Logic Order Search - Trace IN-Clause Option Pop-up</a>" on page 119 for more information.</p>
Action	<p>Specifies how to display the logic order search results. These are the valid actions:</p> <ol style="list-style-type: none"><li>1. Show Subset Name References Only the targets on the path from the starting point to where the logic order search ends are highlighted.</li><li>2. Show Execution Path(s) from Starting Point to Reference(s) All executable statements on the path from the starting point to where the logic order search stops are highlighted.</li><li>3. Trace (single-step) from Starting Point to Reference(s) The logic order search stops at all executable statements on the path from the starting point to where the search ends.</li></ol>

---

## Logic Order Search - Perform Range Name Pop-up

To follow program logic within a perform range, follow this step:

- ▶ Select Logic ▶ Perform and press Enter. The Logic Order Search - Perform Range Name pop-up, shown in [Figure 55](#), displays.

**Figure 55 • Logic Order Search - Perform Range Name Pop-up**

```

Logic Order Search - Perform Range Name

To find "perform range" references in execution order, type the name
and starting point, and select search options. Then press Enter.
For a perform range or label name selection list, type a pattern
(e.g. ABC*) in the appropriate input area.

Perfrange name _____

Start line/label (if any) _____

Direction          Options
1  1. Next          - Trace IN-clause...
   2. Previous
   3. Forward (all)
   4. Backward (all)

Action
1  1. Show perfrange name reference(s)
   2. Show execution path(s) from starting point to reference(s)
   3. Trace (single-step) from starting point to reference(s)
    
```

### Fields

Field	Description
Perfrange name	<p>Specifies the name in a PERFORM statement or the name of any section contained in the DECLARATIVES.</p> <p>To display the Selection List pop-up, leave the Perfrange Name field blank or enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters].</p> <p>See "<a href="#">Selection List Pop-ups</a>" on page 118 for more information.</p>
Start line/label	<p>Specifies the line number or label name to begin the trace.</p> <p>To use cursor position for the beginning of the search, leave this field blank. A message prompting you to verify cursor position displays.</p> <p>To display the Selection List pop-up, enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters].</p> <p>See "<a href="#">Selection List Pop-ups</a>" on page 118 for more information.</p>

Field	Description
Direction	<p>Restricts the logical search to a specific direction or occurrence. These are the valid values:</p> <ol style="list-style-type: none"> <li>1. Next      Searches or traces forward from the current cursor position to the next occurrence of the target. This is the default.</li> <li>2. Previous      Searches or traces backward from the current cursor position to the previous occurrence of the target.</li> <li>3. Forward      Searches or traces forward from the current cursor position to all occurrences of the target.</li> <li>4. Backward      Searches or traces backward from the current cursor position to all occurrences of the target.</li> </ol>
Trace IN-clause	<p>Displays the Logic Order Search - Trace IN-Clause Option pop-up to narrow the scope of the logic order search to a specified perform range or marked set of lines. If not specified, the scope includes the entire program. This option can only be used if number 3 is selected for Action. The default is the entire program is used for the trace.</p> <p>See <a href="#">"Logic Order Search - Trace IN-Clause Option Pop-up" on page 119</a> for more information.</p>
Action	<p>Specifies how to display the logic order search results. These are the valid actions:</p> <ol style="list-style-type: none"> <li>1. Show Perfrange Name References Only the targets on the path from the starting point to where the logic order search ends are highlighted.</li> <li>2. Show Execution Path(s) from Starting Point to Reference(s) All executable statements on the path from the starting point to where the logic order search stops are highlighted.</li> <li>3. Trace (single-step) from Starting Point to Reference(s) The logic order search stops at all executable statements on the path from the starting point to where the search function ends.</li> </ol>

## Logic Order Search - User Mark Name Pop-up

To follow program logic to a specific mark name, follow this step:

Select Logic ► Mark and press Enter. The Logic Order Search - User Mark Name pop-up, shown in [Figure 56](#), displays.

**Figure 56 • Logic Order Search - User Mark Name Pop-up**

```

Logic Order Search - User Mark Name

To find mark name references in execution order, type the name and
starting point, and select search options. Then press Enter. For a
mark or label name selection list, type a pattern (e.g. ABC*) in the
appropriate input area.

Mark name _____

Start line/label (if any) _____

Direction          Options
1  1. Next          _ Trace IN-clause...
   2. Previous
   3. Forward (all)
   4. Backward (all)

Action
1  1. Show mark name reference(s)
   2. Show execution path(s) from starting point to reference(s)
   3. Trace (single-step) from starting point to reference(s)
    
```

### Fields

Field	Description
Mark name	<p>Specifies a 1 to 10 alphanumeric character name given to a set or path using the COPY, MARK, MERGE, or RENAME commands, or one of these system-generated paths:</p> <p>Path:                      Created by:</p> <p>TRACK   TRK                A TRACE command.</p> <p>NETWORK   NET             A FLOW command.</p> <p>SUBNET<sub>n</sub>   SUB<sub>n</sub>         A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (nth result).</p> <p>The TRACK, NETWORK, and SUBNET paths can also be created using the Logic Order Search pop-ups (see "<a href="#">Logic Pull-down</a>" on page 101).</p>

Field	Description
Start line/ label	<p>Specifies the line number or label name at which to begin the trace.</p> <p>To use cursor position for the beginning of the search, leave this field blank. A message prompting you to verify cursor position displays.</p> <p>To display the Selection List pop-up, enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters].</p> <p>See <a href="#">"Selection List Pop-ups" on page 118</a> for more information.</p>
Direction	<p>Restricts the logical search to a specific direction or occurrence by specifying a direction. These are the valid value:</p> <ol style="list-style-type: none"> <li>1. Next            Searches or traces forward from the cursor position to the next occurrence of the target. This is the default.</li> <li>2. Previous       Searches or traces backward from the cursor position to the previous occurrence of the target.</li> <li>3. Forward        Searches or traces forward from the cursor position to all occurrences of the target.</li> <li>4. Backward      Searches or traces backward from the cursor position to all occurrences of the target.</li> </ol>
Trace IN-clause	<p>Displays the Logic Order Search - Trace IN-Clause Option pop-up to narrow the scope of the logic order search to a specified perform range or marked set of lines. If not specified, the scope includes the entire program. This option can only be used if number 3 is selected for Action. The default is the entire program is used for the trace.</p> <p>See <a href="#">"Logic Order Search - Trace IN-Clause Option Pop-up" on page 119</a> for more information.</p>
Action	<p>Specifies how to display the logic order search results. These are the valid actions:</p> <ol style="list-style-type: none"> <li>1. Show Mark Name References <ul style="list-style-type: none"> <li>Only the targets on the path from the starting point to where the logic order search ends are highlighted.</li> </ul> </li> <li>2. Show Execution Path(s) from Starting Point to Reference(s) <ul style="list-style-type: none"> <li>All executable statements on the path from the starting point to where the logic order search stops are highlighted.</li> </ul> </li> <li>3. Trace (single-step) from Starting Point to Reference(s) <ul style="list-style-type: none"> <li>The logic order search stops at all executable statements on the path from the starting point to where the search ends.</li> </ul> </li> </ol>

## Logic Order Search - Subprogram Name Pop-up

To follow program logic in sub-programs, follow this step:

- ▶ Select Logic ▶ Program and press Enter. The Logic Order Search - Subprogram Name pop-up, shown in [Figure 57](#), displays.

**Figure 57 • Logic Order Search - Subprogram Name Pop-up**

```

Logic Order Search - Subprogram Name

To find program name references in execution order, type the name
and starting point, and select search options. Then press Enter.
For a program or label name selection list, type a pattern (e.g.
ABC*) in the appropriate input area.

Program name _____

Start line/label (if any) _____

Direction          Options
1  1. Next          _ Trace IN-clause...
   2. Previous
   3. Forward (all)
   4. Backward (all)

Action
1  1. Show program name reference(s)
   2. Show execution path(s) from starting point to reference(s)
   3. Trace (single-step) from starting point to reference(s)
    
```

### Fields

Field	Description
Program name	<p>Specifies the name of the main program or any nested program. To display the Selection List pop-up, leave the Program Name field blank or enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters].</p> <p>See <a href="#">"Selection List Pop-ups" on page 118</a> for more information.</p>
Start line/label	<p>Specifies the line number or label name from where to begin the trace. To use cursor position for the beginning of the search, leave this field blank. A message prompting you to verify cursor position displays. To display the Selection List pop-up, enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters].</p> <p>See <a href="#">"Selection List Pop-ups" on page 118</a> for more information.</p>

Field	Description
Direction	<p>Restricts the search to a specific direction or occurrence. These are the valid values:</p> <ol style="list-style-type: none"> <li>1. Next      Searches or traces forward from the cursor position to the next occurrence of the target. This is the default.</li> <li>2. Previous      Searches or traces backward from the cursor position to the previous occurrence of the target.</li> <li>3. Forward      Searches or traces forward from the cursor position to all occurrences of the target.</li> <li>4. Backward      Searches or traces backward from the cursor position to all occurrences of the target.</li> </ol>
Trace IN-clause	<p>Displays the Logic Order Search - Trace IN-Clause Option pop-up to narrow the scope of the logic order trace to a specified perform range or marked set of lines. If not specified, the scope includes the entire program. This option can only be used if number 3 is selected for Action. The default is the entire program is used for the trace.</p> <p>See "<a href="#">Logic Order Search - Trace IN-Clause Option Pop-up</a>" on <a href="#">page 119</a> for more information.</p>
Action	<p>Specifies how to display the logic order search results. These are the valid actions:</p> <ol style="list-style-type: none"> <li>1. Show Program Name References <ul style="list-style-type: none"> <li>Only the targets on the path from the starting point to where the logic order search ends are highlighted.</li> </ul> </li> <li>2. Show Execution Path(s) from Starting Point to Reference(s) <ul style="list-style-type: none"> <li>All executable statements on the path from the starting point to where the logic order search function stops are highlighted.</li> </ul> </li> <li>3. Trace (single-step) from Starting Point to Reference(s) <ul style="list-style-type: none"> <li>The logic order search stops at all executable statements on the path from the starting point to where the search ends.</li> </ul> </li> </ol>

## Logic Order Search - Line Number Range Pop-up

To follow program logic within ranges of lines, follow this step:

- ▶ Select Logic ▶ Line and press Enter. The Logic Order Search - Line Number Range pop-up, shown in [Figure 58](#), displays.

**Figure 58 • Logic Order Search - Line Number Range Pop-up**

```

Logic Order Search - Line Number Range

To find line range references in execution order, type the range and
starting point, and select search options. Then press Enter. For a
label name selection list, type a pattern (e.g. ABC*) in the "start"
field.

Line range _____

Start line/label (if any) _____

Direction          Options
1  1. Next          _ Trace IN-clause...
   2. Previous
   3. Forward (all)
   4. Backward (all)

Action
1  1. Show line number range reference(s)
   2. Show execution path(s) from starting point to reference(s)
   3. Trace (single-step) from starting point to reference(s)
    
```

### Fields

Field	Description
Line range	Specifies a single line number or range of lines. Line numbers display in columns 1 through 6 of the Source View.
Start line/label	Specifies the line number or label name from where to begin the trace. To use cursor position for the beginning of the search, leave this field blank. A message prompting you to verify cursor position displays. To display the Selection List pop-up, enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters].  See <a href="#">"Selection List Pop-ups" on page 118</a> for more information.

Field	Description
Direction	<p>Restricts the search to a specific direction or occurrence by specifying a direction. These are the valid values:</p> <ol style="list-style-type: none"> <li>1. Next Searches or traces forward from the cursor position to the next occurrence of the target. This is the default.</li> <li>2. Previous Searches or traces backward from the cursor position to the previous occurrence of the target.</li> <li>3. Forward Searches or traces forward from the cursor position to all occurrences of the target.</li> <li>4. Backward Searches or traces backward from the cursor position to all occurrences of the target.</li> </ol>
Trace IN-clause	<p>Displays the Logic Order Search - Trace IN-Clause Option pop-up to narrow the scope of the logic order trace to a specified perform range or marked set of lines. If not specified, the scope includes the entire program. This option can only be used if number 3 is selected for Action. The default is the entire program is used for the trace.</p> <p>See "<a href="#">Logic Order Search - Trace IN-Clause Option Pop-up</a>" on <a href="#">page 119</a> for more information.</p>
Action	<p>Specifies how to display the logic order search results. These are the valid actions:</p> <ol style="list-style-type: none"> <li>1. Show Line Number Reference(s) <p>Only the targets on the path from the starting point to where the logic order search ends are highlighted. This is the default.</p> </li> <li>2. Show Execution Path(s) from Starting Point to Reference(s) <p>All executable statements on the path from the starting point to where the logic order search stops are highlighted.</p> </li> <li>3. Trace (single-step) from Starting Point to Reference(s) <p>The logic order search stops at all executable statements on the path from the starting point to where the search ends.</p> </li> </ol>

## Selection List Pop-ups

*To display selection lists for dataname, label name, COBOL subset name, perform range name, and program name logic order searches*

- 1 Perform one of these actions:
  - Enter a pattern [question mark (?) for one character; asterisk (\*) for zero or more characters] and press Enter.
  - Leave the Name field blank and press Enter.
  - Select Start line/label field from the various pop-ups and press Enter.

[Figure 59](#) illustrates the Selection List - Data Names pop-up. The selection list pop-ups are the same except for the type of list displayed.

**Figure 59 • Selection List - Data Names Pop-up**

```

Command ==> _____ Selection List - Data Names _____ Scroll ==> CSR
S   Data name
-   -----
-   ABEND-CODE
-   ADDRESS1
-   AREA-CODE
-   AVG-AMT (DEAD)
-   BILLING-DAYS
-   CHECK-CODE (DEAD)
-   CITY
-   CLIENT-ID
-   CONSOLE
-   CUR-PREFIX
-   CURRENT-ZIP-DATA
-   CUSTOMER-ID
-   DBA-DEPT-CODE
-   DEBUG-PARM
-   DET-ADDRESS
-   DET-AREA-CODE
-   DET-BILLING-DAYS
    
```

- 2 Select a dataname by typing S in the line command and press Enter. The selected name is placed in the name field of the Logic Order Search pop-up.

Use the LOCATE primary command to can scroll through the list of names.

## Logic Order Search - Trace IN-Clause Option Pop-up

### *To restrict the source lines considered for a trace*

- 1 From the Logic Order Search pop-up, specify Action 3 - Trace (single-step) from starting point to reference(s) and the Trace IN-clause option, and press Enter. The Logic Order Search - Trace IN-Clause Option pop-up, shown in [Figure 60](#), displays.

**Figure 60 • Logic Order Search - Trace IN-Clause Options Pop-up**

```

Logic Order Search - Trace IN-Clause Option

To restrict the source lines to be considered for a Trace, type one
or more IN-Clause selections. Then press Enter. For a selection
list, type a pattern (e.g. ABC*) in the appropriate input area.

Perform range . . _____
User MARK name . _____

-----
2. Uses only                2. Previous                IN-clause...
3. Mods only                3. Forward (all)
                             4. Backward (all)

Action
3  1. Show data name reference(s)
   2. Show execution path(s) from starting point to reference(s)
   3. Trace (single-step) from starting point to reference(s)

```

- 2 Type the perform range or user mark name to search and press Enter.

To display the Selection List pop-up for the Perform range or the User MARK name fields, follow this step:

- ▶ Enter a pattern with wildcard characters [question mark (?) for one character; asterisk (\*) for zero or more characters] and press Enter.

### Fields

Field	Description
Perform range	Specifies the perform range to search.
User MARK name	Specifies the name of the mark to search.



---

# 6

## Task

---

This chapter describes items on the Insight Task menu and contains these sections:

Topic	Page
<a href="#">Task Pull-down</a>	<a href="#">122</a>
<a href="#">Automated Task - Modify Program Output Pop-up</a>	<a href="#">123</a>
<a href="#">Modify Output Task - Output Selection Pop-up</a>	<a href="#">124</a>
<a href="#">Modify Output Task - Statement Selection Pop-up</a>	<a href="#">126</a>
<a href="#">Modify Output Task - Data Name Selection Pop-up</a>	<a href="#">127</a>
<a href="#">Automated Task - Modify a Computation Pop-up</a>	<a href="#">129</a>
<a href="#">Automated Task - Debug a Program Abend pop-up</a>	<a href="#">134</a>
<a href="#">Automated Task - Result Options Pop-up</a>	<a href="#">137</a>
<a href="#">Options - COPY/Include Libraries Pop-up</a>	<a href="#">138</a>

The Task facility automates program maintenance and enhancement tasks, such as finding the incorrect output source, enhancing computations, and finding the abends source.

Insight shows tasks results in Source View or an edit session.

**Note:**

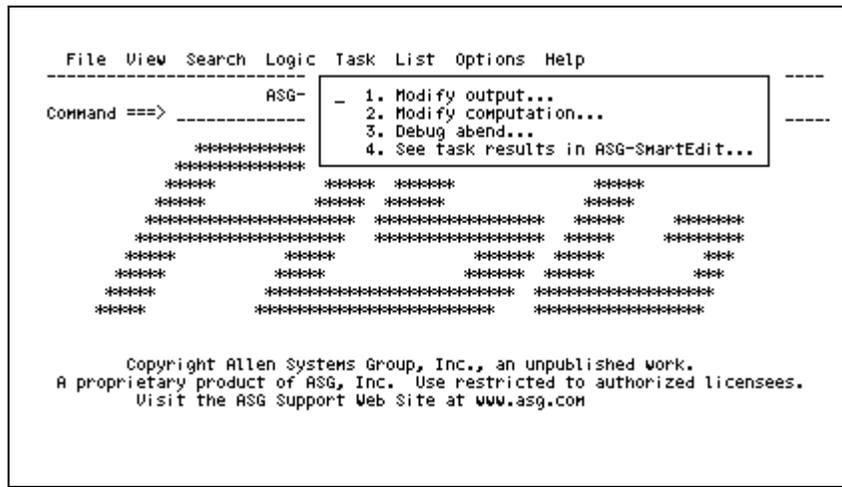
See the *ASG-Insight User's Guide* for more information.

## Task Pull-down

To access the selection pop-ups used to automate common program maintenance and enhancement tasks, follow this step:

- ▶ Select Task on the action bar and press Enter. The Task pull-down, shown in [Figure 61](#), displays.

Figure 61 • Task Pull-down



## Actions

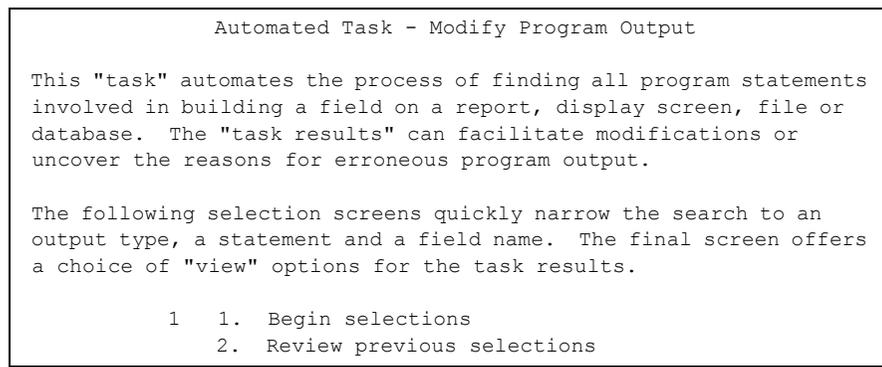
Action	Description
1. Modify Output	Displays the Automated Task - Modify Program Output pop-up used to begin selecting output names for the Modify Program Output task or to review previous selections.
2. Modify Computation	Displays the Automated Task - Modify a Computation pop-up used to begin selecting datanames for the Modify Computation task or to review previous selections.
3. Debug Abend	Displays the Automated Task - Debug a Program Abend pop-up used to begin the Debug Abend task or to review previous selections.
4. See Task Results in the Editor	Displays the Edit Options pop-up used to specify and/or verify options for the editor.  If SmartEdit is installed, this action displays the Edit Options screen for SmartEdit.

## Automated Task - Modify Program Output Pop-up

To begin modifying, correcting, or understanding statements involved in outputting a line on a report or a record on a file, follow this step:

- ▶ Select Task ▶ Modify and press Enter. The Automated Task - Modify Program Output pop-up, shown in [Figure 62](#), displays.

**Figure 62 • Automated Task - Modify Program Output Pop-up**



### Options

Options	Description
Begin selections	Displays the Modify Output Task - Output Selection pop-up used to select the incorrect output to be examined in the Task facility. This is the default.
Review previous selections	Displays the Modify Output Task - Output Selection pop-up with the previously selected output name highlighted.

## Modify Output Task - Output Selection Pop-up

To select the output to examine, follow this step:

- ▶ Select option 1 or 2 on the Automated Task - Modify Program Output pop-up and press Enter. The Modify Output Task - Output Selection pop-up, shown in [Figure 63](#), displays.

**Figure 63 • Modify Output Task - Output Selection Pop-up**

```

Modify Output Task - Output Selection
Command ==> _____ Scroll ==> CSR
The names listed below are the output "files" in the program. Select the
"file" to be investigated. Then press Enter. (Type 'S' to select, 'U' to
un-select, or 'V' to view in Source View.)

Is output performed in CALLED programs (Y/N)? NO

S  Output Name                Type
-----
-  CONSOLE                    DD Name
-  MASTER-RPT                 FD
-  SYSOUT                     DD Name
***** BOTTOM OF DATA *****
    
```

A message indicates the number of files producing program output when this pop-up first displays.

### Fields

Field	Description
Is Output Performed in CALLED Programs	Specifies whether all CALL statements display. Type YES to display all CALL statements for selection. NO is the default
Line command area	Refers to the area to the left of the Output Name field, which accepts these line commands: <ul style="list-style-type: none"> <li>S Selects the output name and displays the Modify Output Task - Statement Selection pop-up to select the statement producing the incorrect output.</li> <li>U Deselects a previously-selected output name. The name is no longer highlighted.</li> <li>V Displays the Source View screen with a shortened action bar, showing the source statement selected for viewing positioned in the middle of the screen, to see the statement in context. The statement is highlighted and tagged to the right with LINE RNG. Use the END command to return to the Modify Output Task - Output Selection pop-up.</li> </ul>

---

Field	Description
Output name	Specifies the file names the program produces output for. When viewing previous selections, the selected name is highlighted.
Type	Indicates the type of output. These are the valid values: <ul style="list-style-type: none"><li>• FD - File description</li><li>• RD - Report description</li><li>• SD - Sort description</li><li>• DD Name - SYSOUT or CONSOLE output</li><li>• CICS - CICS output</li><li>• IMS - CBLTDLI CALL statement</li><li>• CALL - Non-CBLTDLI CALL statement</li><li>• SQL TABLE SQL TABLE declared and/or created in the program</li><li>• SQL VIEW - SQL VIEW declared and/or created in the program</li><li>• IDMS RECORDS - IDMS records referenced in the program</li></ul>

---

## Modify Output Task - Statement Selection Pop-up

To select the statement producing the incorrect output, follow this step:

- ▶ Select an output name on the Modify Output Task - Output Selection pop-up and press Enter. The Modify Output Task - Statement Selection pop-up, shown in [Figure 64](#), displays.

**Figure 64 • Modify Output Task - Statement Selection Pop-up**

```

Modify Output Task - Statement Selection
Command ==> _____ Scroll ==> CSR

The statements listed below "write" to the selected "file". Select the
statement to be investigated. Then press Enter. (Type 'S' to select, 'U'
to un-select, or 'V' to view in Source View.)

Selected output: MASTER-RPT

S Line   Output statement
-----
- 000352 WRITE MAST-RPT FROM DETAIL-LINE1.
- 000357 WRITE MAST-RPT FROM DETAIL-LINE2.
- 000365 WRITE MAST-RPT FROM DETAIL-LINE3.
- 000372 WRITE MAST-RPT FROM DETAIL-LINE4.
- 000424 WRITE MAST-RPT FROM SUB-PRINT.
- 000434 WRITE MAST-RPT FROM RPT-HDG-LINE1.
- 000435 WRITE MAST-RPT FROM RPT-HDG-LINE2.
- 000454 WRITE MAST-RPT FROM TOTAL-PRT
***** BOTTOM OF DATA *****
    
```

The selection list contains all statements that write to the previously-selected output name.

### Fields

Field	Description
Selected output	Specifies the output name that was selected on the Modify Output Task - Output Selection pop-up.
Line command area	<p>Specifies the area to the left of the Line field, which accepts these line commands:</p> <ul style="list-style-type: none"> <li>S Selects the statement and displays the Modify Output Task - Data Name Selection pop-up to select the dataname producing the incorrect output.</li> <li>U Unselects a previously-selected statement. The statement is no longer highlighted.</li> <li>V Displays the Source View screen with a shortened action bar, showing the source statement selected for viewing positioned in the middle of the screen, to see the statement in context. The statement is highlighted and tagged to the right with LINE RNG. Use the END command to return to the Modify Output Task - Statement Selection pop-up.</li> </ul>

Field	Description
Line	Specifies the Source View line number.
Output statement	Specifies the statements corresponding to the selected output. When viewing previous selections, the selected statements is highlighted.

## Modify Output Task - Data Name Selection Pop-up

To select the dataname responsible for the incorrect output, follow this step:

- ▶ Select a statement on the Modify Output Task - Statement Selection pop-up and press Enter. The Modify Output Task - Data Name Selection pop-up, shown in [Figure 65](#), displays.

**Figure 65 • Modify Output Task - Data Name Selection Pop-up**

```

                                Modify Output Task - Data Name Selection
Command ==> _____ Scroll ==> CSR
The names listed below are data items that are output by the "write"
statement selected. Select the item to be investigated. Then press Enter.
(Type 'S' to select, 'U' to un-select, or 'V' to view in Source View.)
Selected statement: 000352 WRITE MAST-RPT FROM DETAIL-LINE1.
S   Data name

```

The selection list contains all datanames referenced on the selected write statement. When this screen first displays a message indicates the number of datanames found.

## Fields

Field	Description
Selected statement	Specifies the statement selected on the Modify Output Task - Statement Selection pop-up.
Line command area	Specifies the area to the left of the Line field, which accepts these line commands:  S Selects the dataname and displays the Automated Task - Result Option pop-up to choose how to display the results of the Task facility.  U Unselects a previously-selected dataname. The name is no longer highlighted.  V View Source. Displays the Source View screen with a shortened action bar, showing the source statement selected for viewing positioned in the middle of the screen, to see the statement in context. The statement is highlighted and tagged to the right with LINE RNG. Use the END command to return to the Modify Output Task - Data Name Selection pop-up.
Data name	Specifies the datanames contained in the selected statement. When viewing previous selections, the selected dataname is highlighted.

## Usage Notes

The Modify Output Task - Data Name Selection pop-up is the last Modify Output Task selection pop-up. After you select a dataname, the Automated Task - Result Options pop-up displays, allowing you to see the results in Source View, modify the results in the editor, or return to the first selection pop-up to review the selections.

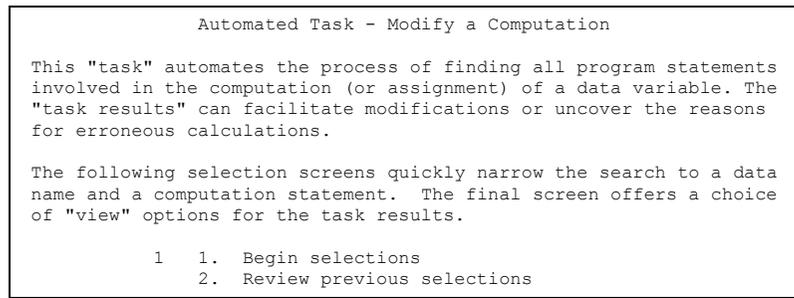
See ["Automated Task - Result Options Pop-up" on page 137](#) for more information.

## Automated Task - Modify a Computation Pop-up

To begin modifying, correcting, or understanding the statements involved in computing or assigning a dataname, follow this step:

- ▶ Select Task ▶ Modify computation and press Enter. The Automated Task - Modify a Computation pop-up, shown in [Figure 66](#), displays.

**Figure 66 • Automated Task - Modify a Computation Pop-up**



### Options

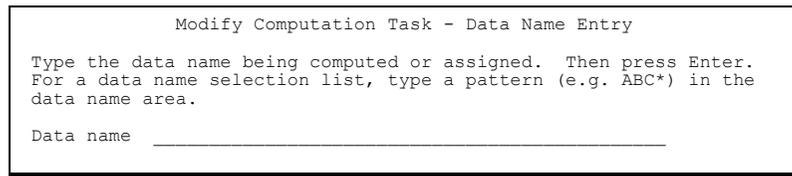
Option	Description
Begin Selections	Displays the Modify Computation Task - Data Name Entry pop-up used to enter the dataname to be examined. This is the default.
Review Previous Selections	Displays the Modify Computation Task - Data Name Selection pop-up with previously selected dataname highlighted.

## Modify Computation Task - Data Name Entry Pop-up

To enter a dataname for examination or to access the Modify Computation Task - Data Name Selection pop-up, follow this step:

- ▶ Select option 1 on the Automated Task - Modify a Computation pop-up and press Enter. The Modify Computation Task - Data Name Entry pop-up, shown in [Figure 67](#), displays.

**Figure 67 • Modify Computation Task - Data Name Entry Pop-up**



The screenshot shows a window titled "Modify Computation Task - Data Name Entry". The text inside the window reads: "Type the data name being computed or assigned. Then press Enter. For a data name selection list, type a pattern (e.g. ABC\*) in the data name area." Below this text is a label "Data name" followed by a horizontal input line.

### Field

Field	Description
Data name	<p>Specifies a COBOL dataname or qualified COBOL dataname. The Modify Computation Task - Statement Selection pop-up displays to specify the statement to examine. Dataname refers to any valid COBOL reference for a data element.</p> <p>To display the Modify Computation Task - Data Name Selection pop-up, leave this field blank or enter a pattern with wildcard characters [question mark (?) for one character; asterisk (*) for zero or more characters].</p>

---

## Modify Computation Task - Data Name Selection Pop-up

To select the dataname to examine, follow this step:

- ▶ Enter a pattern in the Data Name field on the Modify Computation Task - Data Name Entry pop-up and press Enter. The Modify Computation Task - Data Name Selection pop-up, shown in [Figure 68](#), displays.

**Figure 68 • Modify Computation Task - Data Name Selection Pop-up**

```

Command ==>      Modify Computation Task - Data Name Selection      Scroll ==> CSR
The data names listed below are assigned value in this program. Select the
name to be investigated. Then press Enter. (Type 'S' to select, 'U' to
un-select, or 'V' to view in Source View.)

Pattern *
S  Data name          Qualification
-----
- ABEND-CODE
- ADDRESS1           01 MASTER-IN
- AREA-CODE          05 PHONE, 01 MASTER-IN
- BILLING-DAYS       05 LOAN-INFORMATION, 01 MASTER-IN
- CITY               01 MASTER-IN
- CLIENT-ID          01 MASTER-IN
- CUR-PREFIX         01 CURRENT-ZIP-DATA
- CURRENT-ZIP-DATA
- CUSTOMER-ID        05 CLIENT-ID, 01 MASTER-IN

```

The selection list contains all datanames computed or assigned a value in the program for the specified pattern. A message indicates the number of datanames found when this pop-up first displays.

## Fields

Field	Description
Pattern	Specifies the pattern that was used to obtain the list of datanames.
Line command area	Specifies the area to the left of the Line field, which accepts these line commands: <ul style="list-style-type: none"> <li>S Selects the dataname and displays the Modify Computation Task - Statement Selection pop-up to select a statement for examination.</li> <li>U Unselects a previously-selected dataname. The name is no longer highlighted.</li> <li>V View Source. Displays the Source View screen as a pop-up with a shortened action bar, showing the source statement selected for viewing positioned in the middle of the screen to see the statement in context. The statement is highlighted and tagged to the right with LINE RNG. Use the END command to return to the Modify Computation Task - Data Name Selection pop-up.</li> </ul>

Field	Description
Data name	Specifies the datanames included in the pattern that are assigned value in the program. When viewing previous selections, the selected dataname is highlighted.
Qualification	Specifies the parent(s) of the data variable name. The qualifying parent name includes the level number and the dataname.

### Modify Computation Task - Statement Selection Pop-up

*To select a statement for examination and display a list of statements that compute the selected dataname*

- 1 Select a dataname on the Modify Computation Task - Data Name Selection pop-up.

Or

Enter a dataname on the Modify Computation Task - Data Name Entry pop-up.

- 2 Press Enter to display the Modify Computation Task - Statement Selection pop-up shown in [Figure 69](#).

**Figure 69 • Modify Computation Task - Statement Selection**

```

Modify Computation Task - Statement Selection
Command ==> _____ Scroll ==> CSR

The statements listed below assign value to the selected data name.
Select the statement to be investigated. Then press Enter. (Type 'S'
to select, 'U' to un-select, or 'V' to view in Source View.)

Selected name: ABEND-CODE

S Line      Computation/assignment statement
-----
- 000059 77 ABEND-CODE                      PIC S9(4) COMP VALUE +0.
- 000493 MOVE +999 TO ABEND-CODE.
- 000494 CALL 'ABENDPGM' USING ABEND-CODE.
***** BOTTOM OF DATA*****

```

## Fields

Field	Description
Selected name	Specifies the dataname that you entered on the Modify Computation Task - Data Name Entry pop-up or selected on the Modify Computation Task - Data Name Selection pop-up.
Line command area	<p>Specifies the area to the left of the Line field, which accepts these line commands:</p> <ul style="list-style-type: none"> <li>S Selects the statement and displays the Automated Task - Result Options pop-up that allows you to choose how the results of the Task facility display.</li> <li>U Unselects a previously - selected statement. The name is no longer highlighted.</li> <li>V View Source. Displays the Source View screen as a pop-up with a shortened action bar, showing the source statement selected for viewing positioned in the middle of the screen to see the statement in context. The statement is highlighted and tagged to the right with LINE RNG. Use the END command to return to the Modify Computation Task - Statement Selection pop-up.</li> </ul>
Line	Specifies the Source View line number.
Computation/assignment statement	Specifies the statements that compute or assign value to the selected dataname. When viewing previous selections, the selected statements is highlighted.

## Usage Notes

The Modify Computation Task - Statement Selection pop-up is the last Modify Computation Task selection pop-up. After you select a statement, the Automated Task - Result Options pop-up displays, allowing you to see the results in Source View, modify the results in the editor, or return to the first selection pop-up to review the selections.

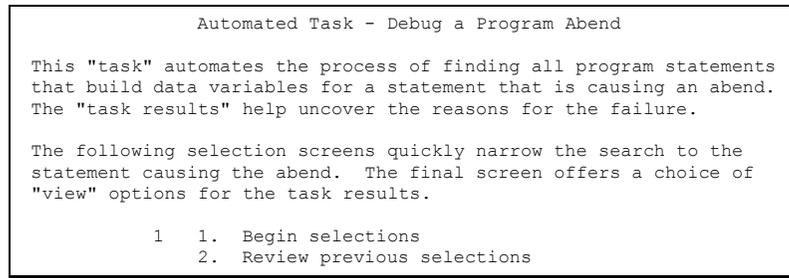
See ["Automated Task - Result Options Pop-up" on page 137](#) for more information.

## Automated Task - Debug a Program Abend pop-up

To begin debugging a data-related abend in a program, follow this step:

- ▶ Select Task ▶ Debug and press Enter. The Automated Task - Debug a Program Abend pop-up, shown in [Figure 70](#), displays.

**Figure 70 • Automated Task - Debug a Program Abend**



The task isolates the statements that compute or assigns values to the data variables in an abending statement.

### Options

Option	Description
Begin selections	Displays the Debug Abend Task - Line Number Entry pop-up used to enter the Source View line number or range of lines where the program abend occurred. This is the default.
Review previous selections	Displays the Debug Abend Task - Line Number Entry pop-up that allows you to review the line number(s) entered.

### Debug Abend Task - Line Number Entry Pop-up

*To enter the Source View line number, or range of lines, where the program abend occurred*

- 1 Select Task ▶ Debug abend and press Enter. The Automated Task - Debug a Program Abend pop-up displays.
- 2 Select option 1 or 2 and press Enter. The Debug Abend Task - Line Number Entry pop-up, shown in [Figure 71 on page 135](#), displays.

- Enter the beginning and ending line where the abend occurred and press Enter. The statements in the range of lines displays.

**Figure 71 • Debug Abend Task - Line Number Entry Pop-up**

```

Debug Abend Task - Line Number Entry

Type the (Source View) line number or the line number range
containing the abending program statement. Then press Enter.

Beginning line number . . 000355

Ending line number . . . _____ (if any)

```

## Fields

Field	Description
Beginning line number	Required. Specifies the Source View line number where the abend occurred, or the beginning line number in a range of lines where the abend occurred.
Ending line number	Specifies the last Source View line number in a range of lines where the abend occurred.

## Debug Abend Task - Statement Selection Pop-up

To select a statement for further examination and to display a list of statements that reference one or more data variables in the specified range of lines, follow this step:

- Enter a line number on the Debug Abend Task - Line Number Entry pop-up. The Debug Abend Task - Statement Selection pop-up, shown in [Figure 72](#), displays.

**Figure 72 • Debug Abend Task Statement Selection Pop-up**

```

Debug Abend Task - Statement Selection
Command ==> _____ Scroll ==> CSR

The statements listed below are in the selected range and reference one or
more data variables. Select the statement to be investigated. Then press
Enter. (Type 'S' to select, 'U' to un-select, or 'V' to view in Source
View.)

Selected lines: 355 -

S Line   Statements containing variables
-----
000355  MOVE YEAR-TO-DATE-INTEREST TO DET-YTD-INT.
***** BOTTOM OF DATA *****

```

## Fields

Field	Description
Selected lines	Specifies the Source View line number(s) entered on the Debug Abend Task - Line Number Entry pop-up.
Line command area	<p>Specifies the area to the left of the Line field, which accepts these line commands:</p> <ul style="list-style-type: none"><li>S Selects the statement and displays the Automated Task - Result Options pop-up to choose how to display the results of the Task facility.</li><li>U Unselects a previously selected statement. The name is no longer highlighted.</li><li>V Displays the Source View screen as a pop-up with a shortened action bar, showing the source statement selected for viewing positioned in the middle of the screen to see the statement in context. The statement is highlighted and tagged to the right with LINE RNG. Use the END command to return to the Modify Computation Task - Statement Selection pop-up.</li></ul>
Line	Specifies the Source View line number.
Statements containing variables	Specifies the statements in the specified range of lines that reference data variables. When viewing previous selections, the selected statements is highlighted.

## Usage Notes

The Debug Abend Task - Statement Selection pop-up is the last Debug Abend Task selection pop-up. After you select a statement, the Automated Task - Result Options pop-up displays, allowing you to see the results in Source View, modify the results in the editor, or return to the first selection pop-up to review the selections.

See ["Automated Task - Result Options Pop-up" on page 137](#) for more information.

## Automated Task - Result Options Pop-up

To choose how the Task facility results display, follow this step:

- ▶ Select dataname (for Modify Program Output Task) or statement (for Modify Computation Task or Debug Abend Task). The Automated Task - Result Options pop-up, shown in [Figure 73 on page 137](#), displays.

**Figure 73 • Automated Task - Result Options Pop-up**

```

Automated Task - Result Options

The selections are complete for this "task". All program statements that
relate to these selections are available for viewing. Select the desired
option. Then press Enter.

1  1. Source View - See the task results in Source View**
   2. Editor      - Modify the results in your editor
   3. Review      - Return to first selection screen

** Source View tags:
    START - Selected starting statement
    IT    - Intermediate target
    TARGET - Statement that initialize component variables
    TARGET-U - Target with an uninitialized variable
    TARGET-U? - Target with possible uninitialized variable

```

### Options

Option	Description
Source View	Redisplays the Source View screen with the statements highlighted and tagged. All other lines are excluded from the display. This is the default option.
Editor	Displays the Options - COPY/Include Libraries pop-up used to access your editor. If SmartEdit is installed, displays the Options - COPY/Include Libraries pop-up for SmartEdit.
Review	Redisplays the first selection list pop-up to review the selections.

## Options - COPY/Include Libraries Pop-up

To specify a source manager for the editor, follow this step:

- ▶ Select Task ▶ See task results in the editor or select Editor on the Automated Task - Result Options pop-up. The Options - COPY/Include Libraries pop-up, shown in [Figure 74](#), displays.

**Figure 74 • Options - COPY/Include Libraries Pop-up**

```

File COBOL SourceManager Options
-----
Options - COPY/Include Libraries
Command ==> _____ Scroll ==> PAGE
Specify Source Manager, COBOL Version and COPY/Include Libraries.
Source Manager . . : PDS/Sequential COBOL Version . : COBOL 74

COPY List:
Data Set Name . . :
Member . . . . . :
Description . . . :

Copy/Include Data Set Name(s) Type VolSer Unit Password
-----
/ 'VIAUSER.TESTD001.CNTL' PDS
***** BOTTOM OF DATA *****
    
```

### Field

Field	Description
Source Manager	Specifies the abbreviation of the source manager that contains your program. These are the valid values: <ol style="list-style-type: none"> <li>1. PDS/Sequential</li> <li>2. Librarian</li> <li>3. Panvalet</li> <li>4. SCLM</li> <li>5. Other (User)</li> </ol>

---

# 7

## List

---

This chapter describes items on the Insight List menu and contains these sections:

Topic	Page
<a href="#">List Pull-down</a>	<a href="#">140</a>
<a href="#">List - CALL Statements Pop-up</a>	<a href="#">141</a>
<a href="#">List - Equates Pop-up</a>	<a href="#">142</a>
<a href="#">List - User Marks Pop-up</a>	<a href="#">143</a>
<a href="#">List - Perform Range Names Pop-up</a>	<a href="#">145</a>
<a href="#">List - Program/Subprogram Names Pop-up</a>	<a href="#">146</a>
<a href="#">List - COBOL Subset Names Pop-up</a>	<a href="#">147</a>



## List - CALL Statements Pop-up

To display programs CALLED by the current program, follow this step:

- ▶ Select List ▶ Calls, or type LIST CALLS on any screen, and press Enter. The List - CALL Statements pop-up, shown in [Figure 76](#), displays.

**Figure 76 • List - CALL Statements Pop-up**

```

List - CALL Statements
Command ==> _____ Scroll ==> CSR
Select the program CALL(s) to be viewed. Then press Enter.
S   Called program                                     Mode
-----
-   'ABENDPGM'                                         STATIC
-   'DBACLSE1'                                         STATIC
-   'DBACLSE2'                                         STATIC
-   'DBAOPEN1'                                         STATIC
-   'DBAOPEN2'                                         STATIC
-   'DBAREAD1'                                         STATIC
-   'DBAREAD2'                                         STATIC
-   'VIAIDEM1'                                         STATIC
***** BOTTOM OF DATA *****

```

A message indicates the number of CALLs listed when this pop-up first displays.

You can copy the CALLED subprograms list to the List file by typing LPRINT \* on this pop-up.

### Fields

Field	Description
S (line command area)	Specifies the area to the left of the CALLED program field, which accepts the S line command that selects a CALLED subprogram for viewing. The screen where you requested the List - CALL Statements pop-up redisplay with the related CALL statements highlighted and tagged.
Called program	Specifies the name of the CALLED module.
Mode	Specifies the type of CALL. These are the valid types: <ul style="list-style-type: none"> <li>STATIC     The object of the CALL is a literal and the associated program is link-edited into the same load module as the CALLing program.</li> <li>DYNAMIC   The object of the CALL is an identifier and is determined at run time. Programs referenced may or may not be link-edited into the same load module as the CALLing program.</li> <li>INTERNAL   A CALL to a nested source subprogram.</li> </ul>

## List - Equates Pop-up

To display all existing equates for the active program, follow this step:

- ▶ Select List ▶ Equates, or type `LIST EQUATES` on any screen, and press Enter. The List - Equates pop-up, shown as [Figure 77](#), displays.

**Figure 77 • List - Equates Pop-up**

```

List - Equates
Command ==> _____ SCROLL ==> CSR
Name      Program      Substitution String      Columns 001-052
-----
MSTR      VIAIDEMO    MASTER-IN + MASTERIN
PARAS     VIAIDEMO    PROGRAM-INIT + P005-VAL-PARM THRU P005-EXIT
ZIPDATA   VIAIDEMO    ZIP-CODE + HLD-ZIP + DET-ZIP-CODE
*****
***** BOTTOM OF DATA *****
    
```

You can copy the Equates list to the List file by typing `LPRINT *` on this pop-up.

## Fields

Field	Description
Name	Specifies the name assigned to the equate.
Program	Specifies the name of the program containing the equate.
Substitution String	Specifies the actual data the NAME represents. Prior to execution, this string is substituted into any command that uses the equate name. This field is left and right scrollable so you can view the entire string. The right-most column contains a greater than (>) symbol when the substitution string extends past the screen column capability. To change the substitution string, type over it.  Type blanks over the substitution string to delete the equate.

## List - User Marks Pop-up

Use the List - User Marks pop-up to perform these functions:

- Display a directory or list of all existing marks
- Change comments
- Merge sets
- Copy sets or paths
- Delete sets or paths
- Rename sets or paths

To display the List - User Marks pop-up, follow this step:

- ▶ Select List ▶ Marks, or type LIST MARKS on any screen, and press Enter. The List - User Marks pop-up, shown in [Figure 78](#), displays.

**Figure 78 • List - User Marks Pop-up**

```

List - User Marks
Command ==> _____ Scroll ==> CSR

C : Copy set/path 1 to set/path 2.   D : Delete existing set/path 1.
M : Merge set/path 1 with set 2.     R : Rename set/path 1 to set/path 2.
S : Select the mark to be viewed.

Set/path 1      Set/path 2      Comments
-----
- NETWORK      PATH _____ FLOW FROM LINE 231 TO 2 TARGET(S) OF 'EXITS'
- SUBNET1      PATH _____ FROM LINE 231 TO TARGET(S) ON LINE 494
- SUBNET2      PATH _____ FROM LINE 231 TO TARGET(S) ON LINE 251
- TRACK        PATH _____ TRACE FROM 231 TO TARGET(S) OF ENTRY
***** BOTTOM OF DATA *****

```

You can copy the Equates list to the List file by typing LPRINT \* on this pop-up.

## Fields

Field	Description
Line command area	Specifies the area to the left of the Set/Path 1 field, which accepts these line commands:  C Copies a set or path. Type the target of the copy in the Set/path 2 field.  M Merges sets. Type the target of the merge in the Set/path 2 field.  S Selects an item for viewing. The screen where you requested the List - User Marks pop-up redisplay with the marks highlighted and tagged.  D Deletes an existing set or path.  R Renames a set or path. Type the new name in the Set/path 2 field.
Set/path 1	Specifies the name of the set or path.
Type	Specifies the type field, between the Set/path 1 field and the Set/path 2 field, identifies the item as a set or path.
Set/path 2	Specifies the appropriate mark name. This field is required if you enter the C, M, or R line command.
Comments	Specifies comment space. Change existing comments by typing over them.

## List - Perform Range Names Pop-up

To display all COBOL perform ranges in the active program, follow this step:

- ▶ Select List ▶ Performs, or type LIST PERFORMS on any screen, and press Enter. The List - Perform Range Names pop-up, shown in [Figure 79](#), displays.

**Figure 79 • List - Perform Range Names Pop-up**

```

List - Perform Range Names
Command ==> _____ SCROLL ==> CSR
Select the perform range to be viewed. Then press Enter.

  Perform range name
  -----
- PROGRAM-INIT
- P000-NEXT THRU P000-EXIT
- P005-VAL-PARM THRU P005-EXIT
- P010-OPEN THRU P019-EXIT
- P100-PRINT THRU P119-EXIT
- P120-READ THRU P129-EXIT
- P150-SUBTOT THRU P169-EXIT
- P155-CL-SUBTOT THRU P159-EXIT
- P160-HDG THRU P169-EXIT
- P999-ABEND-PROGRAM OF ABEND-PROGRAM
***** BOTTOM OF DATA *****

```

A message indicates the number of CALLs listed when this pop-up first displays.

### Fields

Field	Description
S (line command area)	Specifies the area to the left of the Perform range name field, which accepts the S line command to select a perform range for viewing. The screen where you requested the List - Perform Range Names pop-up redisplay with the perform ranges highlighted and tagged.
Perform range name	Specifies the perform label name.

### Usage Notes

Scroll down to view the rest of the list.

You can copy the CALLED subprograms list to the List file by typing LPRINT \* on this pop-up.

## List - Program/Subprogram Names Pop-up

To display the internal subprograms defined within a COBOL II Release 3 or later program, follow this step:

- ▶ Select List ▶ Programs, or type LIST PROGRAMS on any screen, and press Enter. The List - Program/Subprogram Names pop-up, shown in [Figure 80](#), displays.

Figure 80 • List - Program/Subprogram Names Pop-up

```
                                List - Program/Subprogram Names
Command ==> _____ Scroll ==> CSR
Select the program name to be viewed.  Then press Enter.

  Program name
-----
  VIAIDEMO
***** BOTTOM OF DATA *****
```

Internal subprograms are indented in relation to their hierarchy.

When this pop-up first displays a message indicates the number of CALLs listed.

You can copy the CALLED subprograms list to the List file by typing LPRINT \* on this pop-up.

### Fields

Field	Description
S (line command area)	Specifies the area to the left of the Perform range name field, which accepts the S line command to select a subprogram for viewing. The screen where you requested the List - Program/Subprogram Names pop-up redisplay with the subprograms highlighted and tagged.
Program name	Specifies the program or subprogram ID.

## List - COBOL Subset Names Pop-up

To display the subsets along with a brief description of each, follow this step:

- ▶ Select List ▶ Subsets, or type LIST SUBSETS on any screen, and press Enter. The List - COBOL Subset Names pop-up, shown in [Figure 81](#), displays.

**Figure 81 • List - COBOL Subset Names Pop-up**

```

List - COBOL Subset Names
Command ==> _____ Scroll ==> CSR

Select the subset to be viewed. Then press Enter.

S      Subset                Description
-----
_ ASsignment - COBOL: ACCEPT, ADD, SUBTRACT, MULTIPLY, DIVIDE,
              BY, COMPUTE, MOVE, FROM, INSPECT, EXAMINE,
              STRING, UNSTRING, SET, SEARCH, TRANSFORM;
              IDMS: ACCEPT, CHECK, ERASE, GET, INQUIRE, LOAD,
              PUT, RETURN;
              SQL: FETCH, SELECT INTO, SET.
_ CALL      - COBOL: CALL, CANCEL, ENTRY, intrinsic function
              calls;
              CICS: LINK, XCTL;
              IDMS: ATTACH, LINK, XCTL.
_ Cics      - EXEC CICS, EXEC DLI commands.
_ COBOLII   - CALL (COBOLII), PERFORM (COBOLII), CONTINUE, SET,
              SERVICE LABEL, EVALUATE, INITIALIZE, end verbs.
_ COBOL370  - Statements and clauses unique to COBOL/370, such as
              intrinsic function calls, procedure pointers and

```

### Fields

S (line command area)	Specifies the area to the left of the Perform range name field, which accepts the S line command to select a subset for viewing. The screen where you requested the List - COBOL Subset Names pop-up redisplay with the subsets highlighted and tagged.
Subset	Specifies the subset name. See the <i>ASG-Insight User's Guide</i> for a description of each subset type.
Description	Specifies the description of the subset.

### Usage Notes

Scroll to view the rest of the list.

You can copy the CALLED subprograms list to the List file by typing LPRINT \* on this pop-up.



---

# 8

## Options

---

This chapter describes items on the Insight Options menu and contains these sections:

Topic	Page
<a href="#">Options Pull-down</a>	<a href="#">149</a>
<a href="#">Options - Product Parameters Pop-up</a>	<a href="#">151</a>
<a href="#">Options - Product Allocations Pop-up</a>	<a href="#">152</a>
<a href="#">Options - Log/List/Punch Definition Pop-up</a>	<a href="#">154</a>
<a href="#">Options - Script File Allocations Pop-up</a>	<a href="#">157</a>
<a href="#">Options - PF Key Definition Pop-up</a>	<a href="#">158</a>
<a href="#">Options - Processing Modes Pop-up</a>	<a href="#">160</a>
<a href="#">Options - Scratchpad Pop-up</a>	<a href="#">161</a>

### Options Pull-down

You use the Options pull-down to set parameters and options, and to define PF keys. This pull-down also provides access to the scratchpad, and is used to refresh the Source View display. This section describes the actions available on the Options pull-down.



Action	Description
7. Scratchpad	Displays the Options - Scratchpad pop-up used to select options for equates and marks.
8. Refresh	Updates the CALL summaries for programs CALLED by the active program.

## Options - Product Parameters Pop-up

To set the Insight online operation parameters, follow this step:

- ▶ Select Options ▶ Product parameters, or type PARMDEF on any screen, and press Enter. The Options - Product Parameters pop-up, shown in [Figure 83](#), displays.

**Figure 83 • Options - Product Parameters Pop-up**

```

Options - Product Parameters
Command ==> _____
Type the desired parameters below to customize the operation of
the product. Then press Enter.

Alarm . . . . . YES      (Yes or No)
Save Equates . . . . NO   (Yes or No)
Save Marks . . . . . NO   (Yes or No)
Cursor character . . %    (token substitution character)

```

## Fields

Field	Description
Alarm	Controls the audible alarm on the terminal. Type YES for the alarm to sound when an error message displays. Type NO for no alarm. The default is YES.
Save Equates	Saves equates at the end of a Source View session. Type NO if you do not want to save equates. This value can be overridden when using the END, GOBACK, PROGRAM, RETURN, and SAVE commands. The default is YES.  See " <a href="#">EQUATE Command</a> " on page 201 for more information about equates.

Field	Description
Save Marks	<p>Saves marks at the end of a Source View session. Type NO if you do not want to save marks. This value can be overridden when using the END, GOBACK, PROGRAM, RETURN, and SAVE commands. The default is YES.</p> <p>See the <a href="#">"MARK Command" on page 264</a> for more information about user marks.</p>
Cursor Character	<p>Sets the cursor token substitution character. The default is %. Cursor substitution saves time by allowing you to use this character in a command and then place the cursor under the token in the source code.</p> <p>See <a href="#">"Cursor Substitution Character" on page 183</a> for more information.</p>

## Options - Product Allocations Pop-up

To specify the DASD volumes for the Log, List, Punch, and Work files; and to specify Work file space, follow this step:

- ▶ Select Options ▶ Product allocations, or type ALLOCDEF on any screen, and press Enter. The Options - Product Allocations pop-up, shown in [Figure 84](#), displays.

**Figure 84 • Options - Product Allocations Pop-up**

```

Options - Product Allocations
Command ===> _____

Log file:
  Management Class  MGMT1      Generic unit . . . SYSDA
  Storage Class . . STOR1      Volume serial . . _____

List file:
  Management Class  MGMT1      Generic unit . . . SYSDA
  Storage Class . . STOR1      Volume serial . . _____

Punch file:
  Management Class  MGMT1      Generic unit . . . SYSDA
  Storage Class . . STOR1      Volume serial . . _____

Work file:
  Management Class  MGMT1      Generic unit . . . SYSDA
  Storage Class . . STOR1      Volume serial . . _____
  Data Class . . . . DATA1    Space units . . . CYLS
                               Primary space . . 1
                               Secondary space  1
  
```

## Fields

Field	Description
Log File	Specifies the SMS classes or the device type and volume serial number for the Insight Log file. The Log file is used for error messages and log commands. The file characteristics are specified on the Options - Log/List/Punch Definition pop-up. The log file name is <i>userid.INSnnnnn.VIALOG</i> . If the TSO prefix is different from the user ID, the dataset name is preceded by the prefix value.
List File	Specifies the SMS classes or the device type and the volume serial number for the List file, that is allocated the first time you type LPRINT. The List file is used for LPRINT output. The file characteristics are specified on the Options - Log/List/Punch Definition pop-up. The list file name is <i>userid.INSnnnnn.VIALIST</i> . If the TSO prefix is different from the user ID, the dataset name is preceded by the prefix value.
Punch File	Specifies the SMS classes or the device type and the volume serial number for the Punch file, that is allocated the first time you type LPUNCH. The Punch file is used for LPUNCH output. The file characteristics are specified on the Options Log/List/Punch Definition pop-up. The punch file name is <i>userid.INSnnnnn.VIAPUNCH</i> . If the TSO prefix is different from the user ID, the dataset name is preceded by the prefix value.
Work File	Specifies the SMS classes or the device type, volume serial number, and space requirements for the Work file space that is allocated upon entry into Insight. The Work file is a temporary file. Space requirements for it depend on the size of the COBOL programs you work with. Approximately one cylinder is required for a 5,000 line program on a 3380 device.

## Options - Log/List/Punch Definition Pop-up

To set values for allocating, formatting, and processing the Insight Log, List, and Punch files, follow this step:

- ▶ Select Options ▶ Log/list/punch, or type PRINTLOG or PRINTLST on any screen, and press Enter. The Options - Log/List/Punch Definition pop-up, shown in [Figure 85](#), displays.

**Figure 85 • Options - Log/List/Punch Definition Pop-up**

```

Options - Log/List/Punch Definition
Command ==> _____
1 - Process LOG File  2 - Process LIST File  3 - Process PUNCH File

Options                LOG                LIST                PUNCH
-----                ---                ----                -----
Process Option . . . . . PD                PD                PK
Primary tracks . . . . . 1                1                1
Secondary tracks . . . . . 2                5                5
Lines per page . . . . . 56               56               56
Sysout class . . . . . *                *                *

Process Options:  PK (Print/Keep), PD (Print/Delete), K, or D.
LOG FILE IS ALLOCATED

Job Statement Information:
//NAME          JOB (ACCOUNT),NAME
//              MSGCLASS=A
/*              INSERT '/*ROUTE PRINT NODE.USER' HERE IF NEEDED.
//*
```

An internal error allocates the Log file. The List file is allocated the first time you type LPRINT. The Punch file is allocated the first time you type LPUNCH .

The Options - Product Allocations pop-up specifies the Log, List, and Punch file volumes.

## Options

Option	Description
1 - Process LOG File	Verify or change the Options for the Log file and type 1 to process the Log file. If the PK or PD Process Option is specified, type the required Job statement information prior to selecting this option. The Log file is processed when you press Enter. You do not need to exit Insight to print the Log file. By default, the log file name is <code>userid.INSnnnnn.VIALOG</code> . If the TSO prefix is different from the user ID, the dataset name is preceded by the prefix value.
2 - Process LIST File	Verify or change the Options for the List file and type 2 to process the List file. If the PK or PD Process Option is specified, type the required Job statement information prior to selecting this option. The List file is processed when you press Enter. You do not need to exit Insight to print the List file. The list file name is <code>userid.INSnnnnn.VIALIST</code> . If the TSO prefix is different from the user ID, the dataset name is preceded by the prefix value.
3 - Process PUNCH File	Either verify or change the Options for the Punch file and type 3 to process the Punch file. If the PK or PD Process Option is specified, type the required Job statement information prior to selecting this option. The Punch file is processed when you press Enter. You do not need to exit Insight to process the Punch file. Notice that the allocation of the Punch file is used by the Browse facility for determining the space used by the temporary file created during a Browse session. The punch file name is <code>userid.INSnnnnn.VIAPUNCH</code> . If the TSO prefix is different from the user ID, the dataset name is preceded by the prefix value.

## Fields

Field	Description
Process Option	Specifies one of the valid processing options. The default is PD for the Log and List files, and PK for the Punch file.
Primary tracks	Specifies the number of primary tracks to allocate. A size change does not take effect until the next allocation occurs. The default is 1.
Secondary tracks	Specifies the number of secondary tracks to allocate. A size change does not take effect until the next allocation occurs. These are the defaults: 2 for the Log file, 5 for the List and Punch files.

Field	Description
Lines per page	Specifies the number of lines to print per page. Standard maximum values are 60 for 6 lines per inch and 80 for 8 lines per inch. The default is 56.
Sysout class	Specifies the SYSOUT class value. The default is *, that sends the SYSOUT to the destination identified in the MSGCLASS parameter on the JOB statement.
Process Options	Lists the options available for the Log, List, and Punch files. These are the valid values: PK Print and keep K Keep without printing PD Print and delete D Delete without printing
XXXXX FILE IS ALLOCATED	Displays when the Log, List, or Punch file is properly allocated. If the message does not appear, check the assignments on the Options - Product Allocations pop-up. XXXXX is LOG, LIST, or PUNCH depending on the file that was allocated.
Job Statement Information	Specifies the correct job statement information for your installation. These JCL statements are required if the PK or PD process option is specified.

## Options - Script File Allocations Pop-up

To display and/or modify the default script file concatenation sequence for your session, follow this step:

- ▶ Select Options ▶ Script file allocations and press Enter. The Options - Script File Allocations pop-up, shown in [Figure 86](#), displays.

**Figure 86 • Options - Script File Allocations Pop-up**

```

File View Search Logic Task List Options Help
-----
Options - Script File Allocations
Command ==> _____

R - Restore default Script allocations
Enter desired Script file concatenation.
Script file data set names:

_____
_____
'ASG.UTIL.SCRIPT'
```

### Fields

Field	Description
Restore default Script Allocations	Restores the defaults set during the install process.
Script File Data Set Names	Specifies the dataset names for the desired default script files. The files are concatenated in the order entered.

**Note:** \_\_\_\_\_

When initializing for defaults, the lines fill from the bottom to allow concatenation.

### Usage Notes

Default script files are specified for your site when Insight is installed. You can use the Options - Script File Allocations pop-up to modify the default dataset names or the concatenation sequence for the script files. The changes are maintained in your profile.

## Options - PF Key Definition Pop-up

To display and/or redefine PF key values, follow this step:

- ▶ Select Options ▶ PF keys, or type `KEYS` on any Insight screen. The Options - PF Key Definition pop-up, shown in [Figure 87](#), displays.

PF keys 1 through 12 initially display. A similar screen exists for PF keys 13 through 24, that displays by pressing Enter. Any command or data value can be assigned to a PF key.

**Figure 87 • Options - PF Key Definition Pop-up**

```

Options - PF Key (01-12) Definition
Command ==>> _____
Press Enter to process changes and/or to display alternate keys.
Press PF3/15 (END) to exit.

      Number of PF keys: 24      Terminal type: 3278

PF01 HELP
PF02 SPLIT
PF03 END
PF04 RPREF
PF05 RFIND
PF06 FX %
PF07 UP
PF08 DOWN
PF09 SWAP
PF10 BRANCH
PF11 BRANCH BACKUP
PF12 RECALL
    
```

### Fields

Field	Description
Number of PF Keys	Specifies the number of PF keys supported.
Terminal Type	Indicates the type of terminal being used. Insight supports these 3270 type terminals: <ul style="list-style-type: none"> <li>• Model 2 (24 lines x 80 columns)</li> <li>• Model 3 (27 lines x 80 columns)</li> <li>• Model 4 (43 lines x 80 columns)</li> <li>• Model 5 (27 lines x 133 columns)</li> </ul>
PF01 through PF12 or PF13 through PF24	Specifies the values assigned to the PF keys, that you can change by typing over the number. To reset a PF key to its default value, delete the entry and press Enter. To set a PF key to have no value (i.e., to turn off a PF key) type <code>NOF</code> .

**Usage Notes**

When Insight is installed, PF01 through PF12 and PF13 through PF24 are set to the default values for the primary PF keys. These are the default values:

Value	Description
PF01/13	HELP
PF07/19	UP
PF02/14	SPLIT
PF08/20	DOWN
PF03/15	END
PF09/21	SWAP
PF04/16	RPREF
PF10/22	BRANCH
PF05/17	RFIN D
PF11/23	BRANCH BACKUP
PF06/18	FX %
PF12/24	RECALL

**Note:**

The alternate PF keys may be PF01 through PF12 or PF13 through PF24, depending on your site's specifications.

## Options - Processing Modes Pop-up

### To set processing modes

- 1 Select Options ► Modes and press Enter. The Options - Processing Modes pop-up, shown in [Figure 88](#), displays.
- 2 Change the settings by typing over them.

You can also change the setting using the SET command, for example, by typing SET LEARN ON. See "[SET Command](#)" on page 311 for more information.

**Figure 88 • Options - Processing Modes Pop-up**

```

Options - Processing Modes
Command ==> _____ Scroll ==> CSR
Type option settings. Then press PF3/15 (END) to save options and exit.
-----
Option      Set      Description
-----
LEARN       OFF     Display internally generated Primary Commands
SCRIPT      OFF     Script facility is disabled
XMODE       OFF     Exclude all lines before executing Primary Commands
    
```

### Fields

Field	Description
Option	The processing mode to be enabled or disabled.
LEARN	When LEARN is ON, all primary commands generated internally when actions from the pull-downs are executed display on the LEARN Mode - Generated Command pop-up. See the " <a href="#">SET Command</a> " on page 311 for more information. The default for this option is OFF.
SCRIPT	When SCRIPT is ON, a script file is allocated. From that point on, all Insight primary commands are captured in this file. When SCRIPT is OFF, the file is closed and saved. The script file name is <i>userid.INSnnnnn.VIASCRIPT</i> . If the TSO profile prefix is not the same as the user ID, then this name is preceded by a prefix value.
XMODE	When XMODE is ON, all lines are excluded from the screen before a primary command is executed. The default is OFF.

Field	Description
Set	Displays current setting for each mode. You can change the settings by typing over them. Values not set to the default status are shown in bold.
Description	Provides descriptions of the options. The Script facility description also indicates whether it is enabled or disabled.

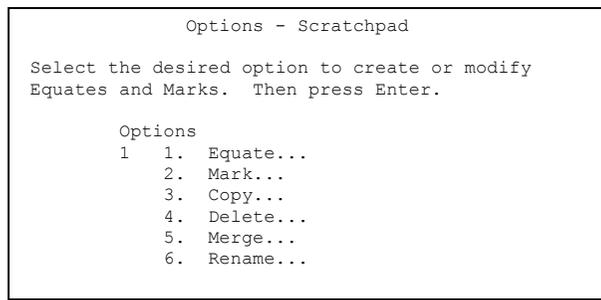
## Options - Scratchpad Pop-up

Use the Options - Scratchpad pop-up to define equates and to create, copy, delete, merge, or rename marks.

To display the Options - Scratchpad pop-up, follow this step:

- Select Options ► Scratchpad and press Enter. The Options - Scratchpad pop-up, shown in [Figure 89](#), displays.

**Figure 89 • Options - Scratchpad Pop-up**



## Options

Option	Description
Equate	Displays the Options - Scratchpad Equate pop-up used to define a unique name representing a character string or group of data items. See <a href="#">"Options - Scratchpad Equate Pop-up" on page 162</a> for more information.
Mark	Displays the Options - Scratchpad Mark pop-up used to save the requested target as a mark set or path. See <a href="#">"Options - Scratchpad Mark Pop-up" on page 164</a> for more information.

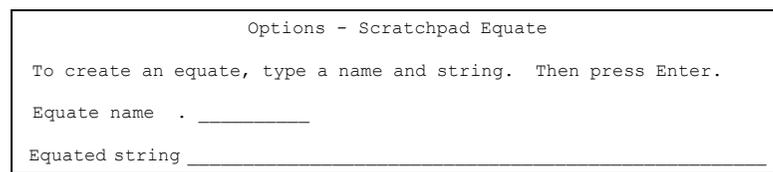
Option	Description
Copy	Displays the Options - Scratchpad Copy pop-up used to copy a mark set or path. See " <a href="#">Options - Scratchpad Copy Pop-up</a> " on page 165 for more information.
Delete	Displays the Options - Scratchpad Delete pop-up used to delete a mark set or a path. See " <a href="#">Options - Scratchpad Delete Pop-up</a> " on page 167 for more information.
Merge	Displays the Options - Scratchpad Merge pop-up used to add the lines from the specified target to the specified mark name. See " <a href="#">Options - Scratchpad Merge Pop-up</a> " on page 168 for more information.
Rename	Displays the Options - Scratchpad Rename pop-up used to change the name of a mark path or a set of lines. See " <a href="#">Options - Scratchpad Rename Pop-up</a> " on page 169 for more information.

### **Options - Scratchpad Equate Pop-up**

To define a character string name, follow this step:

- ▶ Select Equate on the Options - Scratchpad pop-up and press Enter. The Options - Scratchpad Equate pop-up, shown in [Figure 90](#), displays.

**Figure 90 • Options - Scratchpad Equate Pop-up**



The screenshot shows a dialog box titled "Options - Scratchpad Equate". The text inside reads: "To create an equate, type a name and string. Then press Enter." Below this text are two input fields: "Equate name . \_\_\_\_\_" and "Equated string \_\_\_\_\_".

**Fields**

Fields	Description
Equate name	<p>Specifies the name that you want to assign to the character string. Names must conform to these standards:</p> <ul style="list-style-type: none"><li>• 1 to 10 alphanumeric characters</li><li>• First character must be alphabetic or a period</li><li>• A hyphen is the only special character allowed</li><li>• May be 1 to 4 DBCS characters</li></ul>
Equated string	<p>Specifies a character string to be substituted by the equate. A character string can be a long command, pattern, dataname, or concatenated dataname, etc. You can specify literals and blanks in the character string if desired. The string may be a DBCS string. If the equated string is not entered, the equate name is deleted.</p>

## Options - Scratchpad Mark Pop-up

To save the requested target as a mark set or path, follow this step:

- ▶ Select Options ▶ Mark and press Enter. The Options - Scratchpad Mark pop-up, shown in [Figure 91](#), displays.

**Figure 91 • Options - Scratchpad Mark Pop-up**

```

Options - Scratchpad Mark

To create a mark, type a name and a target. Then press Enter. For a
name selection list for a target type, type a pattern (e.g. ABC*) in
the target name area.

Mark name . . _____

Target name . _____

Target type
1  1. Mark name
   2. Perfrange name
   3. Program name
   4. Subset name
   5. Line range
   6. Label name

Comments . . _____
    
```

## Fields

Field	Description
Mark name	Specifies the name you want to assign to the set of lines or path. The name must be 1 to 10 alphanumeric characters.
Target name	Specifies the name of the target you want to save.
Target type	Specifies the type of target you want to save.
Mark name	<p>A 1 to 10 alphanumeric character name given to a set or path using the COPY, MARK, MERGE, or RENAME commands, or one of these system-generated paths:</p> <p>System generated path: Created by:</p> <p>TRACK   TRK      A TRACE command.</p> <p>NETWORK   NET    A FLOW command.</p> <p>SUBNET<sub>n</sub>   SUB<sub>n</sub>    A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (nth result).</p>

Field	Description
	<p><b>Note:</b></p> <p>The TRACK, NETWORK, and SUBNET paths can also be created using the Logic Order Search pop-ups (see <a href="#">"Logic Pull-down" on page 101</a>).</p>
Perfrange Name	The name specified in a PERFORM statement including all statements executed as a result of a PERFORM statement, or the name of any section contained in the Declaratives.
Program Name	The name of the main program or any nested program representing all the code contained in the program. This includes all the programs physically nested inside the specified program.
Subset Name	A predefined COBOL language subset. See the <i>ASG-Insight User's Guide</i> for a list of each subset and its description.
Line Range	A single line number or range of lines. Line numbers display in columns 1 through 6 of Source View.
Label Name	Any PROCEDURE DIVISION paragraph name or section name. The PROCEDURE and PROC literals can also be specified.
Comments	Provides a description or other comments for the mark.

### Options - Scratchpad Copy Pop-up

To copy the contents of a path or set of lines to a mark, follow this step:

- ▶ Select Copy on the Options - Scratchpad pop-up and press Enter. The Options - Scratchpad Copy pop-up, shown in [Figure 92](#), displays.

**Figure 92 • Options - Scratchpad Copy Pop-up**

```

Options - Scratchpad Copy

To create a mark, type a name and a target. Then press Enter. For a
name selection list for a target type, type a pattern (e.g. ABC*) in
the target name area.

Mark name . . _____
Target name . _____

Target type
1  1. Mark name
   2. Perfrange name
   3. Program name
   4. Subset name
   5. Line range
   6. Label name

Comments . . _____

```

## Fields

Field	Description								
Mark name	Specifies the name you want to assign to the set of lines or path. The name must be 1 to 10 alphanumeric characters.								
Target name	Specifies the name of the target you want to save.								
Target type	Specifies the type of target you want to save.								
Mark name	<p>Specifies a 1 to 10 alphanumeric character name given to a set or path using the COPY, MARK, MERGE, or RENAME commands, or one of these system-generated paths:</p> <table> <tbody> <tr> <td>System generated path:</td> <td>Created by:</td> </tr> <tr> <td>TRACK   TRK</td> <td>A TRACE command.</td> </tr> <tr> <td>NETWORK   NET</td> <td>A FLOW command.</td> </tr> <tr> <td>SUBNET<sub>n</sub>   SUB<sub>n</sub></td> <td>A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (nth result).</td> </tr> </tbody> </table> <p><b>Note:</b> _____  The TRACK, NETWORK, and SUBNET paths can also be created using the Logic Order Search pop-ups (see <a href="#">"Logic Pull-down" on page 101</a>).</p>	System generated path:	Created by:	TRACK   TRK	A TRACE command.	NETWORK   NET	A FLOW command.	SUBNET <sub>n</sub>   SUB <sub>n</sub>	A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (nth result).
System generated path:	Created by:								
TRACK   TRK	A TRACE command.								
NETWORK   NET	A FLOW command.								
SUBNET <sub>n</sub>   SUB <sub>n</sub>	A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (nth result).								
Perfrange name	Specifies the name in a PERFORM statement, including all of the statements executed as a result of a PERFORM statement, or the name of any section contained in the Declaratives.								
Program name	Specifies the name of the main program or of any nested program representing all of the code contained in the program. This includes all of the programs physically nested inside the specified program.								
Subset name	Specifies a predefined COBOL language subset. See the <i>ASG-Insight User's Guide</i> for a list of each subset and its description.								
Line range	Specifies a single line number or a range of lines. Line numbers display in columns 1 through 6 of Source View.								
Label name	Specifies any PROCEDURE DIVISION paragraph name or section name. You can also specify the PROCEDURE and PROC literals.								
Comments	Provides a description or other comments for the mark.								

## Usage Notes

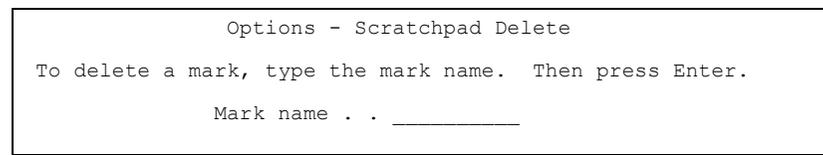
You can also copy paths and sets using the COPY command. See the ["COPY Command" on page 194](#) for more information.

## Options - Scratchpad Delete Pop-up

To delete a mark set or path, follow this step:

- ▶ Select Delete on the Options - Scratchpad pop-up and press Enter. The Options - Scratchpad Delete pop-up, shown in [Figure 93](#), displays.

**Figure 93 • Options - Scratchpad Delete Pop-up**



## Field

Field	Description
Mark name	Specifies the name of the mark you want to delete.

## Usage Notes

You can also delete marks using the DELETE command. See the ["DELETE Command" on page 198](#) for more information.

## Options - Scratchpad Merge Pop-up

To add the lines from the specified target to the specified mark name, follow this step:

- ▶ Select Merge on the Options - Scratchpad pop-up and press Enter. The Options - Scratchpad Merge pop-up, shown in [Figure 94](#), displays.

**Figure 94 • Options - Scratchpad Merge Pop-up**

```

Options - Scratchpad Merge

To expand a mark, type the name and target. Then press Enter. For a
name selection list for a target type, type a pattern (e.g. ABC*) in
the target name area.

Mark name . . _____
Target name . _____

Target type
1  1. Mark name
   2. Perfrange name
   3. Program name
   4. Subset name
   5. Line range
   6. Label name

Comments . . _____
    
```

## Fields

Field	Description						
Mark name	Specifies the name you want to assign to the new set of lines or path. The name must be 1 to 10 alphanumeric characters.						
Target name	Specifies the name of the target you want to add.						
Target type	Specifies the type of target you want to add.						
Mark name	<p>A 1 to 10 alphanumeric character name given to a set or path using the COPY, MARK, MERGE, or RENAME commands, or one of these system-generated paths:</p> <p>System generated path: Created by:</p> <table border="0"> <tr> <td>TRACK   TRK</td> <td>A TRACE command.</td> </tr> <tr> <td>NETWORK   NET</td> <td>A FLOW command.</td> </tr> <tr> <td>SUBNET<sub>n</sub>   SUB<sub>n</sub></td> <td>A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (nth result).</td> </tr> </table>	TRACK   TRK	A TRACE command.	NETWORK   NET	A FLOW command.	SUBNET <sub>n</sub>   SUB <sub>n</sub>	A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (nth result).
TRACK   TRK	A TRACE command.						
NETWORK   NET	A FLOW command.						
SUBNET <sub>n</sub>   SUB <sub>n</sub>	A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (nth result).						

**Note:**

The TRACK, NETWORK, and SUBNET paths can be created using the Logic Order Search pop-ups (see ["Logic Pull-down" on page 101](#)).

Field	Description
Perfrange name	The name specified in a PERFORM statement, including all statements executed as a result of a PERFORM statement, or the name of any section contained in the Declaratives.
Program name	The name of the main program or of any nested program representing all of the code contained in the program. This includes all of the programs physically nested inside the specified program.
Subset name	A predefined COBOL language subset. See the <i>ASG-Insight User's Guide</i> for a list of each subset and its description.
Line range	A single line number or a range of lines. Line numbers display in columns 1 through 6 of Source View.
Label Name	Any PROCEDURE DIVISION paragraph name or section name. You can also specify the PROCEDURE and PROC literals.
Comments	Provides a description or other comments for the mark.

### Usage Notes

You can also merge marks using the MERGE command. See the ["MERGE Command" on page 267](#) for more information.

### Options - Scratchpad Rename Pop-up

To change the name of a mark path or set of lines, follow this step:

- ▶ Select Rename on the Options - Scratchpad pop-up and press Enter. The Options - Scratchpad Rename pop-up, shown in [Figure 95](#), displays.

**Figure 95 • Options - Scratchpad Rename Pop-up**

Options - Scratchpad Rename

To rename a mark, type the old and new name. Then press Enter.

Old mark name \_\_\_\_\_

New mark name \_\_\_\_\_

## Fields

Field	Description
Old mark name	Specifies the mark name that you want to change.
New mark name	Specifies the new name that you want to assign to the mark. The name must be 1 to 10 alphanumeric characters.

## Usage Notes

Marks can also be renamed with the RENAME command. See the ["RENAME Command" on page 289](#) for more information.

---

# 9

## Help

---

This chapter describes items on the Insight Help menu options and contains these sections:

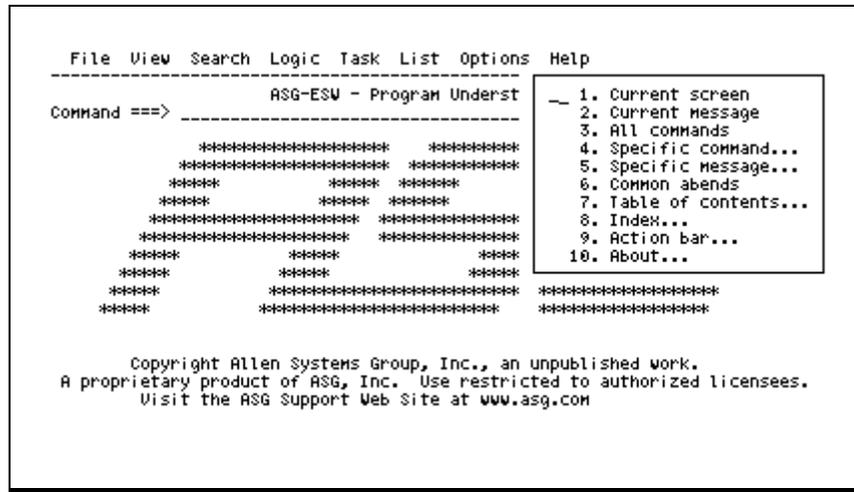
Topic	Page
<a href="#">Help Pull-down</a>	<a href="#">172</a>
<a href="#">Help - Specific ASG Command Pop-up</a>	<a href="#">174</a>
<a href="#">Help - Specific ASG Message Number Pop-up</a>	<a href="#">175</a>
<a href="#">Help - About Pop-up</a>	<a href="#">175</a>

## Help Pull-down

To access the online help facility, follow this step:

- ▶ Select Help on the action bar and press Enter. The Help pull-down, shown in [Figure 96](#), displays.

Figure 96 • Help Pull-down



## Actions

Action	Description
1. Current Screen	Displays help for the screen or pop-up that is currently displayed.
2. Current Message	Displays help for the message that is currently displayed.
3. All Commands	Displays a complete list of all Insight primary commands, where specific command information displays by selecting the appropriate number.
4. Specific Command	Displays the Help - Specific ASG Command pop-up used to obtain help about a specific Insight primary command.
5. Specific Message	Displays the Help - Specific ASG Message Number pop-up used to obtain help about a specific message number.
6. Common Abends	Displays the Abends screen, where you can display information about a specific abend by selecting the appropriate number. Select topic 2 on this screen to display the ASG Abend Codes screen, that lists all the user abends and their explanations.

---

Action	Description
7. Table of Contents	Displays the Help Table of Contents used to request help for general information. See the <a href="#">"Help Facility" on page 391</a> for more information and an example of the online Help Table of Contents.
8. Index	Displays the Help Index used to request help for specific information. See the <a href="#">"Help Facility" on page 391</a> for more information and an example of the online Help Index.
9. Action Bar	Displays the Action Bar help pop-up that contains general information about the action bar and selections to help for each of the pull-downs.
10. About	Displays the Help - About pop-up that lists information about the currently installed levels of Insight and Center.

### Usage Notes

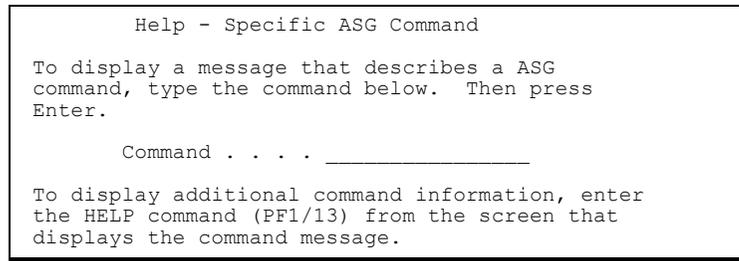
You can also use the HELP command to access help. See the ["HELP Command" on page 234](#) for more information.

## Help - Specific ASG Command Pop-up

To obtain help about a specific Insight primary command, follow this step:

- ▶ Select Help ▶ Specific ASG Command and press Enter. The Help - Specific ASG Command pop-up, shown in [Figure 97](#), displays.

**Figure 97 • Help - Specific ASG Command Pop-up**



### Fields

Field	Description
Command	Specifies a primary command. The Help Tutorial for that command displays. See the <a href="#">"Help Facility" on page 391</a> for more information.

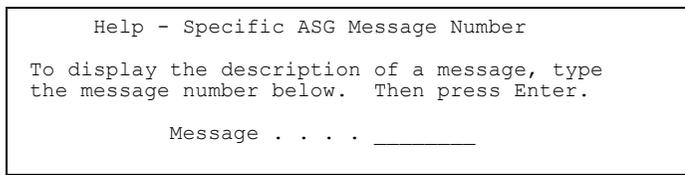
---

## Help - Specific ASG Message Number Pop-up

To display help for a specific messageASG, follow this step:

- ▶ Select Help ▶ Specific ASG Message Number and press Enter. The Help - Specific ASG Message Number pop-up, shown in [Figure 98](#), displays.

**Figure 98 • Help - Specific ASG Message Number Pop-up**



### Fields

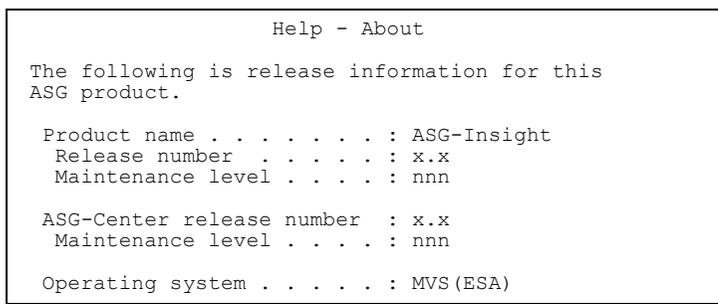
Field	Description
Message	Specifies a message number. The HELP Explanation and Action Panel for that message displays. See the <a href="#">"Help Facility" on page 391</a> for more information.

## Help - About Pop-up

To display information about the currently installed releases of Insight and Center, follow this step:

- ▶ Select Help ▶ About and press Enter. The Help - About pop-up, shown in [Figure 99](#), displays.

**Figure 99 • Help - About Pop-up**



## Fields

Field	Description
Product name	Specifies Insight or the name of the ESW product you are currently running.
Release number	Specifies the currently installed release number of Insight.
Maintenance level	Specifies the currently installed Insight PRS number.
ASG-Center release number	Specifies the currently installed release number of Center.
Maintenance level	Specifies the currently installed Center PTF number.
Operating system	Specifies the operating system you are running.

**Note:** \_\_\_\_\_  
This information is requested when you contact the ASG Service Desk for assistance. You can also display this information by typing `PRODLVL` on any screen.  
\_\_\_\_\_

---

---

# 10

## Commands

---

This chapter describes Insight commands and contains these sections:

Topic	Page
<a href="#">Command Processing</a>	<a href="#">180</a>
<a href="#">Recalling/Repeating Commands</a>	<a href="#">181</a>
<a href="#">Cursor Position</a>	<a href="#">182</a>
<a href="#">Cursor Substitution Character</a>	<a href="#">183</a>
<a href="#">Command Diagrams</a>	<a href="#">183</a>
<a href="#">&amp; (Retain) Command</a>	<a href="#">186</a>
<a href="#">ALLOCDEF Command</a>	<a href="#">187</a>
<a href="#">ANALYZE Command</a>	<a href="#">188</a>
<a href="#">BRANCH Command</a>	<a href="#">189</a>
<a href="#">CALL Command</a>	<a href="#">192</a>
<a href="#">CANCEL Command</a>	<a href="#">193</a>
<a href="#">COPY Command</a>	<a href="#">194</a>
<a href="#">CURRENT Command</a>	<a href="#">197</a>
<a href="#">DELETE Command</a>	<a href="#">198</a>
<a href="#">END Command</a>	<a href="#">200</a>
<a href="#">EQUATE Command</a>	<a href="#">201</a>
<a href="#">EXCLUDE Command</a>	<a href="#">203</a>

Topic	Page
<a href="#">EXECUTE Command</a>	<a href="#">209</a>
<a href="#">FIND Command</a>	<a href="#">212</a>
<a href="#">FINDXTND Command</a>	<a href="#">216</a>
<a href="#">FLOW Command</a>	<a href="#">228</a>
<a href="#">GOBACK Command</a>	<a href="#">233</a>
<a href="#">HELP Command</a>	<a href="#">234</a>
<a href="#">HIGH Command</a>	<a href="#">235</a>
<a href="#">JUMP Command</a>	<a href="#">242</a>
<a href="#">KEYS Command</a>	<a href="#">243</a>
<a href="#">LEVELS Command</a>	<a href="#">245</a>
<a href="#">LIST Command</a>	<a href="#">246</a>
<a href="#">LOCATE Command</a>	<a href="#">248</a>
<a href="#">LPRINT Command</a>	<a href="#">250</a>
<a href="#">LPUNCH Command</a>	<a href="#">257</a>
<a href="#">MARK Command</a>	<a href="#">264</a>
<a href="#">MERGE Command</a>	<a href="#">267</a>
<a href="#">PARMDEF Command</a>	<a href="#">270</a>
<a href="#">PREF Command</a>	<a href="#">271</a>
<a href="#">PRINTLOG Command</a>	<a href="#">274</a>
<a href="#">PRINTLST Command</a>	<a href="#">275</a>
<a href="#">PROCESS Command</a>	<a href="#">276</a>
<a href="#">PRODLVL Command</a>	<a href="#">280</a>
<a href="#">PROGRAM Command</a>	<a href="#">281</a>

Topic	Page
<a href="#">QUALIFY Command</a>	<a href="#">282</a>
<a href="#">RECALL Command</a>	<a href="#">283</a>
<a href="#">REDO Command</a>	<a href="#">286</a>
<a href="#">REFRESH Command</a>	<a href="#">288</a>
<a href="#">RENAME Command</a>	<a href="#">289</a>
<a href="#">REPEAT Command</a>	<a href="#">290</a>
<a href="#">RESET Command</a>	<a href="#">291</a>
<a href="#">RETURN Command</a>	<a href="#">292</a>
<a href="#">RFIND Command</a>	<a href="#">293</a>
<a href="#">RHIGH Command</a>	<a href="#">294</a>
<a href="#">RPREF Command</a>	<a href="#">295</a>
<a href="#">RSCROLL Command</a>	<a href="#">296</a>
<a href="#">RSTRUCTURE-VIEW Command</a>	<a href="#">297</a>
<a href="#">RTRACE Command</a>	<a href="#">298</a>
<a href="#">RTREEVW Command</a>	<a href="#">301</a>
<a href="#">SAVE Command</a>	<a href="#">302</a>
<a href="#">SCROLL Command</a>	<a href="#">303</a>
<a href="#">SELECT Command</a>	<a href="#">310</a>
<a href="#">SET Command</a>	<a href="#">311</a>
<a href="#">SOURCEVIEW Command</a>	<a href="#">312</a>
<a href="#">STRUCTURE-VIEW Command</a>	<a href="#">313</a>
<a href="#">TRACE Command</a>	<a href="#">315</a>
<a href="#">TREEVIEW Command</a>	<a href="#">321</a>

Topic	Page
<a href="#">VIEW Command</a>	<a href="#">323</a>
<a href="#">ZOOMIN/ZOOMOUT Commands</a>	<a href="#">324</a>
<a href="#">Line Commands</a>	<a href="#">325</a>

In addition to processing requests from pull-downs and pop-ups, Insight accepts the primary and line commands that you enter in the same manner as ISPF commands. You can enter primary commands on screens and pop-ups that contain a command input area; line commands are entered in a selection column field or in the prefix area over the line numbers in Source View. Insight supports all ISPF system commands on the appropriate screens. All Insight commands are described in this chapter.

Some of the most frequently used commands have been assigned to PF keys. The KEYS command can be entered from any command input area to change the PF key assignments. See the online help for detailed information about PF key assignments.

## Command Processing

Insight line commands process before primary commands. Commands you enter by pressing a PF key are handled as if you entered them in the command input area. If you enter a command by pressing a PF key and there is a command in the command input area, the contents of the command input area append to the PF key command. The combined commands then execute as a whole. Multiple commands entered in the command input area are separated by a semicolon (;) and process in a left-to-right sequence. Command results display when one of these conditions occur:

- The last command processes and all commands in the sequence complete successfully.
- A command is not successfully processed. If an error occurs in the sequence, processing stops and an error message displays. Results display for the commands that successfully completed. The command causing the error and all remaining commands in the sequence display in the command input area.

Each command in a command sequence is recognized separately for use with the RECALL command.

## Recalling/Repeating Commands

Insight remembers the last twenty primary commands entered. These methods can be used to repeat commands:

- Type `RECALL` to display the last command entered in the command input area. After the recalled command displays, it can be modified, deleted, or executed again. Enter the `RECALL` command repeatedly to display the commands in reverse sequence without executing them. To execute a recalled command, press Enter while the command displays in the command input area.
- Type `REPEAT` to execute the last stacked primary command again (if it is a repeatable command).

These are the Insight primary commands that can be repeated:

- `BRANCH`
- `CANCEL`
- `COPY`
- `DELETE`
- `EQUATE`
- `EXCLUDE`
- `EXECUTE`
- `FIND`
- `FINDXTND`
- `FLOW`
- `HIGH`
- `LIST`
- `LPRINT`
- `LPUNCH`
- `MARK`
- `MERGE`
- `PREF`
- `REFRESH`
- `RENAME`
- `RESET`
- `RPREF`
- `SCROLL`
- `SELECT`
- `SET`
- `SOURCEVIEW`
- `STRUCTURE-VIEW`
- `TRACE`
- `TREEVIEW`
- `ZOOMIN`
- `ZOOMOUT`

These methods can be used to repeat the last primary command entered:

- Use the & (Retain) command preceding a primary command to keep the command displayed in the command input area after execution. This is the quickest way to repeat a command when minor changes are required before execution.
- Use the REDO command to re-execute the last FIND, FINDXTND, HIGH, PREF, SCROLL, or TRACE command (see ["REDO Command" on page 286](#) for more information).
- Use the RFIND command to repeat the last FIND or FINDXTND command from the cursor position. PF05/17 is the default PF key for the RFIND command.
- Use the RHIGH command to repeat the last HIGH command from the cursor position.
- Use the RPREF command to display the last View - Paragraph Cross Reference pop-up or to function as a PREF command with the cursor location as the target and default direction (see ["RPREF Command" on page 295](#) for more information). PF04/16 is the default PF key for the RPREF command.
- Use the RSCROLL command to repeat the last SCROLL command from the cursor position.
- Use the RTRACE command to continue the last TRACE command from where it stopped (see ["RTRACE Command" on page 298](#) for more information). PF06/18 is the default PF key for the RTRACE command.

## Cursor Position

The result of the command depends on the starting point of the search. You can specify a starting point by entering a line number or a label name, or by using the cursor as the starting point. These are the rules if the cursor is used as the starting point:

If cursor is in...	The search is started...
Command input area	In the first column of the first source line on the screen
Line command area	From the first column on that line
Program source	From the cursor position

## Cursor Substitution Character

The cursor character is a token substitution character that you can use in any primary command on the Source View screen. A token is a contiguous set of characters preceded by and followed by a blank, period, comma, or parenthesis. You define the cursor substitution character on the Options - Product Parameters pop-up. You may choose any character, but it should be one that is unique and rarely used in commands. The default is percent (%).

The cursor substitution character saves you time and typing effort when you are working with commands. Type the cursor substitution character anywhere on the command line, then place the screen cursor on the token that is to replace it in the command. For example, to find all occurrences of HLD-ZIP-PREFIX, type `FINDXTND` in the command input area:

```
FX %
```

Then place the screen cursor anywhere on HLD-ZIP-PREFIX in the source code. The command `FX HLD-ZIP-PREFIX` executes when you press Enter. Notice that the cursor substitution character must have a space before and after it.

You can use multiple cursor tokens to specify consecutive tokens in the source code. For example:

```
FX % % %
```

If you then place the screen cursor on HLD-ZIP-PREFIX, the next two tokens are picked up in the command, resulting in the command `FX HLD-ZIP-PREFIX OF HLD-ZIP`. The first token is HLD-ZIP-PREFIX, the second token is OF, and the third token is HLD-ZIP. The tokens must be consecutive.

## Command Diagrams

These are the notational conventions and symbols used to describe command syntax:

Item	Description
ABBREVIations	Illustrates the command abbreviation, which is shown in uppercase letters. Lowercase letters in the command are optional.
lowercase	Indicates user-supplied variable information.
UPPERCASE	Indicates commands or keywords.

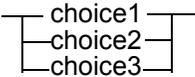
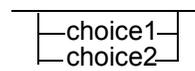
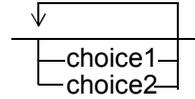
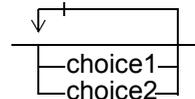
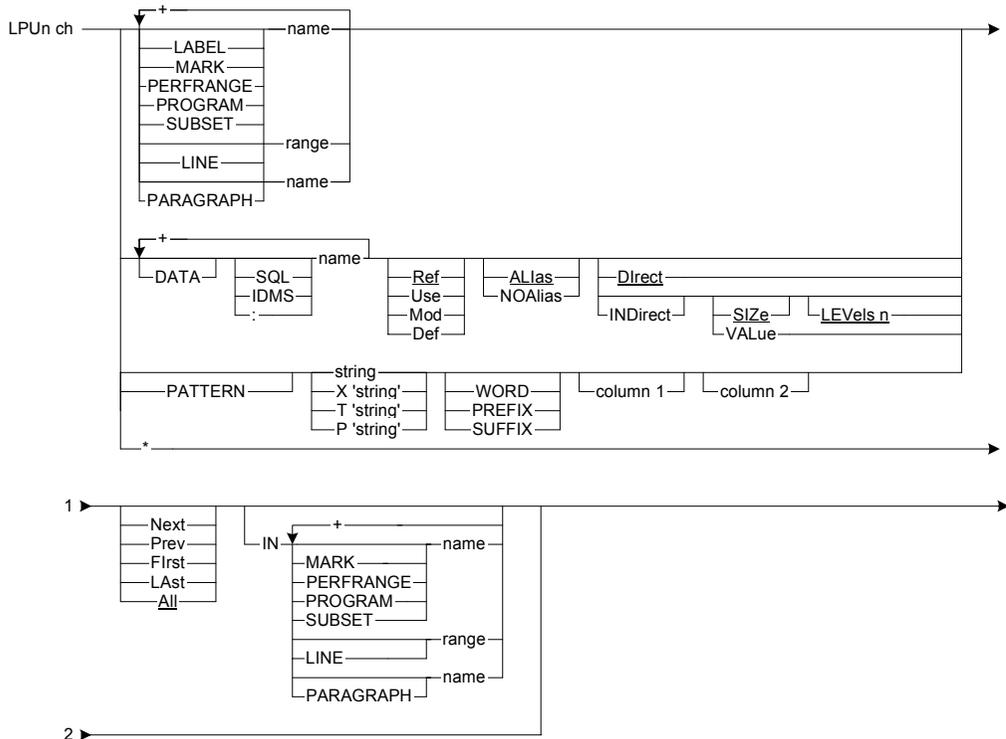
Item	Description
<b>Bold</b>	Indicates operands that are available only if Insight is installed and a Insight analysis has been run on the COBOL program being tested.
<u>Underline</u>	Specifies the default value of an operand.
	Separates synonymous commands or operands.
—————>	Indicates that the command syntax is continued on the next line.
—————>	Indicates the command syntax is continued from the previous line.
—————×	Indicates the end of the command syntax.
— required —	Indicates that the operand or keyword appearing on the main command line is required.
	Indicates that one operand is required.
	Indicates that an operand or keyword appearing below the main command line is optional.
	Indicates that operands are optional.
	Indicates that more than one operand can be chosen.
	Indicates that operands can be concatenated by placing a plus sign (+) between them.

Figure 100 illustrates a syntax diagram for the LPUNCH command. LPUNCH or a minimum of LPU can be entered. You must enter LPUNCH with a target operand as indicated by the name variable on the main path of the line. If the path is followed down the first or second vertical line, you can specify any of the target operands or an asterisk (\*). If you specify more than one operand, separate each with a space. Operands that can be concatenated are indicated by a returning arrow that includes a plus sign (+) in the line. Dataname operands (Ref, Use, etc.) pertain to all datanames in a concatenated series.

Continuation lines are numbered 1 and 2 with 1 being the main path of the line. Continuation line 2 shows that the asterisk (\*) operand is entered with no other values since the line from it extends to the end of the path, bypassing the direction and intarget operands. The direction and intarget operands are optional since they are below the main path of the line.

Figure 100 • Command Syntax Diagram



## & (Retain) Command

&any primary command 

### Function

The Retain command executes the specified primary command and keeps it displayed in the command input area for repeated use or modification.

### Operands

Ampersand (&) must be followed by a primary command.

### Usage Notes

The Retain command is useful if you want to repeatedly execute the same primary command or make minor changes to a command.

For example, after you execute the FINDXTND command for a data item, you may desire to execute an LPRINT command on the same data item. You could perform this function quickly and easily by using the retain command.

### Example

To find the EOF-FLAG and keep &FINDXTND EOF-FLAG displayed in the command input area, type this command:

```
&FINDXTND EOF-FLAG
```

## ALLOCDEF Command

ALLOCDEF | ADEF

---

### *Function*

The ALLOCDEF command displays the Options - Product Allocations pop-up used to specify the DASD volumes for the Log, List, Punch, and Work files; and to specify space for the Work file.

### *Operands*

None.

### *Usage Notes*

The Options - Product Allocations pop-up can also be displayed by selecting Options ▶ Product allocations and pressing Enter.

You can enter the ALLOCDEF command on any screen.

## ANALYZE Command

ANalyze 

### Function

The ANALYZE command displays the File - Analyze Submit pop-up used to submit a compile/analyze job without ending the current Insight function. A program must be analyzed before it can be used in Insight.

### Operands

None.

### Usage Notes

The File-Analyze Submit pop-up can also be displayed by selecting File ► Analyze and pressing Enter.

You can use the ANALYZE command on any Insight screen. If viewing a program in Source View and a program is encountered that has not been analyzed, perform these steps:

#### *To analyze a program*

- 1 Type ANALYZE in the command input area.
- 2 Submit the analyze job for the program.
- 3 Type END in the command input area.

TSO displays a message when the analyze job has completed. Use the CALL, VIEW, or PROGRAM command to view the program.

#### **Note:**

See ["Analyze" on page 339](#) for additional information about submitting an analyze job. See the online help for more information about the File - Analyze Submit pop-up.

---

## BRANCH Command



### Function

The BRANCH command is used to position the cursor at the specified target. This command is used to scroll from a statement such as a PERFORM, to the paragraph being performed. Use the BACKUP operand to return to the statement where the BRANCH occurred.

### Operands

Operand	Description
Blank	No operand is required if the cursor is positioned on a GO TO or PERFORM statement. The BRANCH command automatically locates the target of the GO TO, PERFORM, or internal CALL statement. Internal CALL statements are the CALLs in a COBOL II Release 3 program that call programs located in the same source module. If the cursor is positioned on any other statement, the BRANCH command locates the next sequential statement to be executed. The PROCEDURE DIVISION label is located if the cursor is positioned outside the PROCEDURE DIVISION.
PERFRANGE <i>name</i>	Locates the specified PERFORM range. If the specified name is not fully qualified and there are multiple PERFORM ranges with the specified name, the cursor is positioned on the first PERFORM range found. A message displays indicating the number of PERFORM ranges found.
PROGRAM <i>name</i>	Locates the specified nested program. For COBOL II Release 3 or later programs, a data item, label, or program name that may be ambiguous or used multiple times can be qualified by using OF followed by the program name. For example, if the label P120-READ exists in another program and also in the program VIAIDEM1, it can be qualified as P120-READ OF VIAIDEM1.

Operand	Description
LABEL <i>name</i>	Locates the specified paragraph or section name. If the specified name is not fully qualified and there are multiple paragraphs or sections with the specified name, the cursor is positioned on the first paragraph or section found. A message displays indicating the number of paragraphs or sections found.
PARAGRAPH <i>name</i>	Locates the specified paragraph name. If the specified name is not fully qualified and there are multiple paragraphs with the specified name, the cursor is positioned on the first paragraph found. A message displays indicating the number of paragraphs found.
BACKup	Positions the cursor where the Branch occurred.
PREVious	Works in conjunction with the current cursor location to position the cursor at the GO TO, PERFORM, or CALL statement where a target was reached.

**Note:** \_\_\_\_\_

If more than one statement transfers control to the specified branch location (or cursor position), the Branch Previous Options pop-up displays.

### Usage Notes

The Branch function can also be initiated from the Search - Branch Request pop-up. To display the Search - Branch Request pop-up select Search ► Branch and press Enter. See the online help for more information.

The BRANCH command is very helpful when you are tracking branching logic. This command can be used to track branching logic several levels deep into PERFORMed code, then return to each PERFORM statement. An effective way to use the BRANCH command is to position the cursor on a PERFORM or GO TO statement, then press PF10/22. (BRANCH is the default setting for the PF10/22 key; BRANCH BACKUP is the default for the PF11/23 key.) The screen position is saved for use with the BACKUP operand based on these conditions:

- A PERFORM range or LABEL name is entered on the command line.
- The cursor is placed on a GO TO or PERFORM statement.

The PROCEDURE DIVISION label displays when you enter the BRANCH command with no operands and the cursor is positioned in a part of the program other than the PROCEDURE DIVISION.

If a PROGRAM EXIT or STOP RUN is encountered, a pause occurs and a message displays indicating that a logical end is reached. Press PF10/22 to continue the BRANCH command.

## Examples

**Example 1.** This example assumes that the cursor is positioned on PERFORM READ-ALL. The screen is scrolled and the cursor is positioned on the READ-ALL paragraph.

```
BRANCH
```

**Example 2.** This example assumes that the cursor is positioned on the READ-ALL paragraph. The screen is scrolled and the cursor is positioned on the PERFORM READ-ALL statement.

```
BRANCH PREVIOUS
```

**Example 3.** The screen is scrolled and the cursor is positioned on the P120-READ paragraph.

```
B P120-READ
```

**Example 4.** The screen is scrolled and the cursor is positioned on the PROGRAM-INIT paragraph.

```
B PROGRAM-INIT
```

**Example 5.** The screen is scrolled and the cursor is positioned at the location from where the B PROGRAM-INIT command was entered.

```
B BACKUP
```

## CALL Command



### Function

The CALL command is used to start a Source View session for another program without ending the current session. The GOBACK or END command can be used to return to the original program.

### Operands

Operand	Description
Blank	Displays the AKR Program Selection pop-up for selection of a program. See the online help for more information.
<i>pgm</i>	Specifies a COBOL program or entry point name. If the specified program or entry point is not available in the current AKR, an error message displays.

### Usage Notes

The CALL command provides you a means of following logic paths across CALLED programs. CALL remembers the location within the CALLING program and displays the CALLED program. When finished with the CALLED program, use the GOBACK command to return to the location in the CALLING program. See the ["GOBACK Command" on page 233](#) and the ["PROGRAM Command" on page 281](#) for additional information.

### Examples

**Example 1.** In this example the current Source View program is VIAIDEMO, this command calls the VIAIDEM1 program:

```
CALL VIAIDEM1
```

**Example 2.** Returns to the VIAIDEMO Source View session without saving marks or equates that you created in VIAIDEM1:

```
GOBACK VIAIDEMO NOSAVE
```

## CANCEL Command

CANcel 

### *Function*

The CANCEL command terminates the current function without saving changes to marks and equates made during the session.

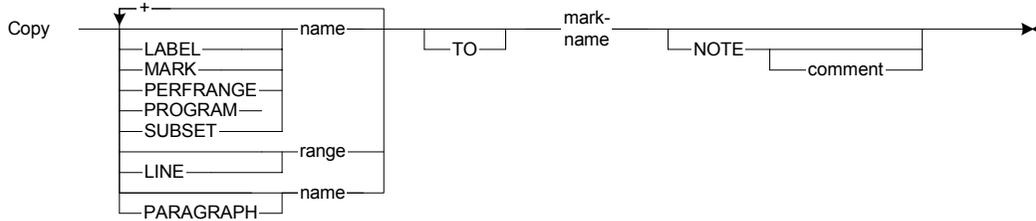
### *Operands*

None.

### *Usage Notes*

CANCEL is the equivalent of an END NOSAVE command. It terminates the function without saving new marks or equates, or changes to existing marks or equates.

## COPY Command



### Function

The COPY command is used to copy the contents of a path or set of lines to a mark. This allows you to save the same information under a different name for additional use. You can also include a new description if desired.

### Operands

Operand	Description								
LABEL <i>name</i>	Specifies a PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. All transfers of control to the label name are included.								
MARK <i>name</i>	Specifies a 1 to 10 alphanumeric character name given to a set or path using the COPY, MARK, MERGE, or RENAME commands, or one of these system-generated paths: <table border="0" style="margin-left: 20px;"> <tr> <td>System generated path:</td> <td>Created by:</td> </tr> <tr> <td>TRACK   TRK</td> <td>A TRACE command.</td> </tr> <tr> <td>NETWORK   NET</td> <td>A FLOW command.</td> </tr> <tr> <td>SUBNET<sub><i>n</i></sub>   SUB<sub><i>n</i></sub></td> <td>A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (<i>n</i>th result).</td> </tr> </table>	System generated path:	Created by:	TRACK   TRK	A TRACE command.	NETWORK   NET	A FLOW command.	SUBNET <sub><i>n</i></sub>   SUB <sub><i>n</i></sub>	A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results ( <i>n</i> th result).
System generated path:	Created by:								
TRACK   TRK	A TRACE command.								
NETWORK   NET	A FLOW command.								
SUBNET <sub><i>n</i></sub>   SUB <sub><i>n</i></sub>	A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results ( <i>n</i> th result).								
<p><b>Note:</b> _____</p> <p>The TRACK, NETWORK, and SUBNET paths can also be created using the Logic Order Search pop-ups (see <a href="#">"Logic Pull-down" on page 101</a>).</p>									

Operand	Description
PERFRANGE <i>name</i>	Specifies the name in a PERFORM statement including all statements executed as a result of a PERFORM statement, or the name of any section contained in the Declaratives.
PROGRAM <i>name</i>	Specifies the name of either the main program or any nested program. All code contained in the specified program is used.
SUBSET <i>name</i>	Specifies a predefined COBOL language subset. See the <i>ASG-Insight User's Guide</i> for a list of each subset and its description.
LINE <i>range</i>	Specifies a single line number or range of lines. Line numbers display in columns 1 through 6 of Source View.
PARAGRAPH <i>name</i>	Specifies a PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. The entire paragraph or section is included.
TO	Specifies an optional keyword indicating the mark name that follows is where the target is copied.
<i>mark-name</i>	Specifies a name assigned to a path or set of lines using the MARK command or the Options - Scratchpad Mark pop-up. The name you specify must meet these requirements: <ul style="list-style-type: none"> <li>• A maximum of 10 alphanumeric characters, including hyphens. Names longer than 10 characters truncate.</li> <li>• Must begin with an alphabetic character.</li> <li>• This name is assigned to the new path or set</li> </ul>
NOTE comment	Provides an optional description about the name. Comment text can be a maximum of 50 alphanumeric or 23 DBCS characters. If you do not enter NOTE comment, the existing comment is copied to the new mark name. If you enter NOTE with no comment text, the existing comment is not copied to the new mark name.

### Usage Notes

You can also use the Options - Scratchpad Copy pop-up to copy paths and sets. See the online help for more information.

The same path or set of lines can be copied to different names. The TO *mark-name* you specify must be unique. If you specify an existing *mark-name*, an error message displays and the copy function is not performed.

To concatenate targets, place a plus sign (+) between the target names. For example, IO+CALL. You can use a concatenated set name wherever a set name is valid. These sets can be concatenated:

- LABEL
- MARK
- PERFRANGE
- SUBSET
- LINE
- PROGRAM
- PARAGRAPH

For COBOL II Release 3 or later programs, you can qualify a data item, label, or program name that may be ambiguous or used multiple times by using OF followed by the program name. For example, if the label P120-READ exists in another program and also in the program VIAIDEM1, you can qualify it as P120-READ OF VIAIDEM1.

### *Examples*

**Example 1.** This command copies the contents of TRACK into FICA:

```
COPY TRACK TO FICA
```

**Example 2.** This command copies all highlighted lines to MYSET:

```
COPY HI TO MYSET
```

## CURRENT Command

CURRent 

### *Function*

The CURRENT command is used to save the current location of the cursor. The LOCATE &CURRENT command can then be used to reposition the screen to the saved location.

Only one CURRENT command can be in effect. A new CURRENT command replaces the prior one.

**Note:** \_\_\_\_\_

The CURRENT command is valid only on the Source View screen.

---

### *Operands*

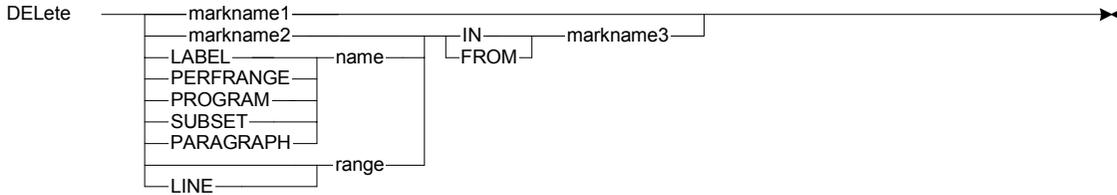
None.

### *Usage Notes*

The CURRENT and LOCATE &CURRENT commands increase the ability to control the position in the source during script processing.

Use the CURRENT command to save your current location while you browse related data items.

## DELETE Command



### Function

The DELETE command is used to delete a mark name and its contents. This command is also used to delete a mark name, path, or a set of lines from an existing mark name.

### Operands

Operand	Description
<i>markname1</i> , <i>markname2</i> , <i>markname3</i>	<p>Specifies a name assigned to a path or set of lines using the COPY, MERGE, MARK, or RENAME commands, or one of these system-generated paths:</p> <p>System generated path:      Created by:</p> <p>TRACK   TRK                      A TRACE command.</p> <p>NETWORK   NET                    A FLOW command.</p> <p><b>Note:</b> _____</p> <p>The TRACK and NETWORK paths can also be created using the Logic Order Search pop-ups (see <a href="#">"Logic Pull-down" on page 101</a>).</p>
LABEL <i>name</i>	Specifies a PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. All transfers of control to the label name are also deleted.
PERFRANGE <i>name</i>	Specifies the name in a PERFORM statement including all statements executed as a result of a PERFORM statement, or the name of any section contained in the Declaratives.
PROGRAM <i>name</i>	Specifies the name of the main program or any nested program representing all the code contained in the program. This includes all the programs physically nested inside the specified program.
SUBSET <i>name</i>	Specifies a predefined COBOL language subset. See the <i>ASG-Insight User's Guide</i> for a description of each subset.

Operand	Description
PARAGRAPH <i>name</i>	Specifies a PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. The entire paragraph or section is deleted.
LINE <i>range</i>	Specifies a single line number or range of lines. Line numbers display in columns 1 through 6 of Source View.
IN/FROM <i>markname3</i>	Deletes the specified mark name, path, or set of lines from the existing <i>markname3</i> .

### Usage Notes

You can delete mark names that you no longer need by using the DELETE command. If you attempt to delete a mark name that does not exist, an error message displays. You can also delete the NETWORK and TRACK system-generated paths. SUBNETs are automatically deleted when you delete the corresponding NETWORK.

#### Note:

To delete a mark name and its contents, select Options ► Scratchpad, then select Delete from the Options - Scratchpad pop-up. See the online help for more information about the Options - Scratchpad pop-up.

The set of lines in one mark name can be deleted from another mark name. In addition, a path or line range can be deleted from an existing mark name.

### Examples

**Example 1.** This command deletes the mark name FICA and its contents:

```
DELETE FICA
```

**Example 2.** This command deletes line 17 from the mark name FICA:

```
DELETE 17 FROM FICA
```

**Example 3.** This command deletes the set of lines in the mark name FICA from the mark name WITHHOLD:

```
DELETE FICA FROM WITHHOLD
```

## END Command



### Function

The END command is used to terminate the current function and to redisplay the previous screen.

### Operands

Operand	Description
Blank	If you enter the END command with no operands, the values specified on the Options - Product Parameters pop-up are used to determine if marks and equates are to be saved.
Save	Saves marks and equates created for this program in the AKR. This operand overrides the values set on the Options - Product Parameters pop-up.
Nosave	Specifies that the marks and equates created for this program are not saved. This operand overrides the values set on the Options - Product Parameters pop-up.

### Usage Notes

The END command (default PF03/15) terminates any Insight function. While in Source View, you can enter the command and the desired operand in the command input area, or you can enter the operand in the command input area and press the PF key. The previous function or screen displays.

## EQUATE Command

Equate —name  →

### Function

The EQUATE command is used to define a name for a character string.

### Operands

Operand	Description
<i>name</i>	<p>Specifies a name for the character string. Names must conform to these standards:</p> <ul style="list-style-type: none"> <li>• 1 to 10 alphanumeric characters</li> <li>• First character must be alphabetic or a period</li> <li>• A hyphen is the only special character allowed</li> <li>• The name may also be 1 to 4 DBCS characters</li> </ul>
<i>string</i>	<p>Specifies a character string to be substituted by the EQUATE command. A character string can be a long command, pattern, dataname, concatenated dataname, etc. You can specify literals and blanks in the character string if desired. You must enclose character strings that include blanks within single or double quotes. The string may be a DBCS string.</p> <p>If you do not enter the string operand, the equated name is deleted</p>

### Usage Notes

You can define equates on the Options - Scratchpad Equate pop-up, and delete them from both the Options - Scratchpad Equate and the Options - Scratchpad Delete pop-ups. Change the substitution string on the List - Equates pop-up by typing over it with the new value.

Use equated names to reduce the number of keystrokes during a Source View session. You can use multiple equates if desired.

If you use an equated name in another EQUATE command, you must first define it. For example:

```
EQ CC TOT-COST
EQ BB CC + PROD-COST
```

BB now contains TOT-COST + PROD-COST

You can save the equates that you created in Source View in the AKR. See the ["SAVE Command" on page 302](#) for more information. To delete an equated name, type `EQUATE` and the name without the string.

**Note:** \_\_\_\_\_

See the online help for more information about the Options - Scratchpad Equate pop-up, the Options - Scratchpad Delete pop-up, and List - Equates pop-up.

---

## Examples

**Example 1.** This command equates the name A to EOF-FLAG:

```
EQ A 'EOF-FLAG'
```

To find EOF-FLAG, type this command:

```
FIND A
```

**Example 2.** This command equates AA to TOT-COST:

```
EQ AA TOT-COST
```

To change AA to be TOT-COST \* PROD-COST, type this command:

```
EQ AA * PROD-COST.
```

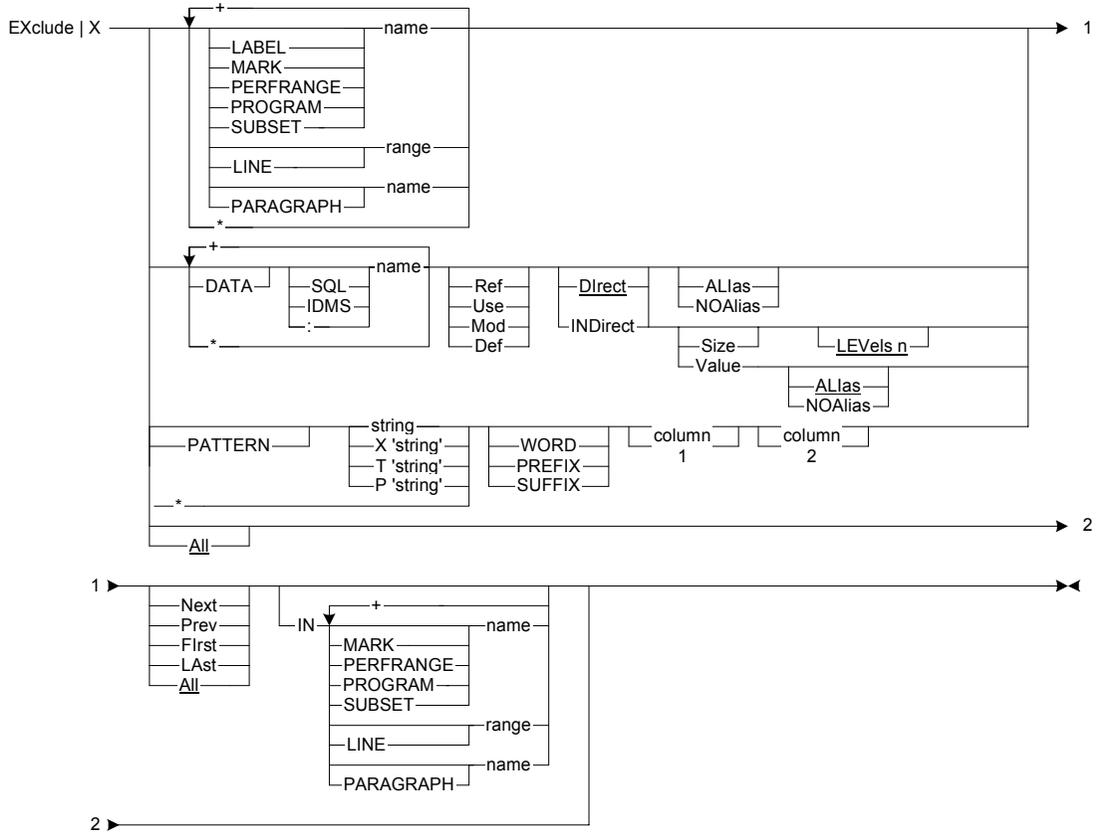
**Example 3.** This command equates BB to the FIND 'END-OF-FILE-FLAG' command:

```
EQ BB FIND 'END-OF-FILE-FLAG'  
BB to FIND 'END-OF-FILE-FLAG'
```

This command deletes the BB equate:

```
EQ BB
```

# EXCLUDE Command



## Function

The EXCLUDE command performs a FINDXTND command on the specified target, excluding the resulting lines.

## Operands

Operand	Description
LABEL <i>name</i>	Specifies a PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. All transfers of control to the label name are included.
MARK <i>name</i>	Specifies a 1 to 10 alphanumeric character name given to a set or path using the COPY, MARK, MERGE, or RENAME commands, or one of these system-generated paths:

Operand	Description
	<p>System generated path:      Created by:</p> <p>TRACK   TRK                      A TRACE command.</p> <p>NETWORK   NET                    A FLOW command.</p> <p>SUBNET<sub>n</sub>   SUB<sub>n</sub>                A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (n<sup>th</sup> result).</p>
	<p><b>Note:</b></p> <p>The TRACK, NETWORK, and SUBNET paths can also be created using the Logic Order Search pop-ups (see <a href="#">"Logic Pull-down" on page 101</a>).</p>
PERFRANGE <i>name</i>	<p>Specifies the name in a PERFORM statement including all statements executed as a result of a PERFORM statement, or the name of any section contained in the Declaratives.</p> <p>This operand is valid in Source View only</p>
PROGRAM <i>name</i>	<p>Specifies the name of the main program or any nested program representing all the code contained in the program. This includes all the programs physically nested inside the specified program.</p>
SUBSET <i>name</i>	<p>Specifies a predefined COBOL language subset. See the <i>ASG-Insight User's Guide</i> for a list of each subset and its description.</p>
LINE <i>range</i>	<p>Specifies a single line number or range of lines. Line numbers display in columns 1 through 6 of Source View.</p>
PARAGRAPH <i>name</i>	<p>Specifies a PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. The entire paragraph or section is included.</p>
asterisk (*)	<p>Reuses the target of the previous search, exclude, find, highlight, or scroll action or command.</p>
DATA <i>name</i>	<p>Specifies a COBOL dataname or qualified COBOL dataname. Dataname refers to any valid COBOL reference for a data element. These subordinate operands apply to the dataname and can be used to further qualify the EXCLUDE command: REF, USE, MOD, DEF, ALIAS, NOALIAS, DIRECT, and INDIRECT. The defaults are REF, ALIAS, and DIRECT.</p>
SQL <i>name</i>	<p>Excludes datanames that are DB2/SQL variables only.</p>

Operand	Description
IDMS <i>name</i>	Excludes datanames that are IDMS variables only.
: <i>name</i>	Excludes datanames that are COBOL variables only.
Ref	Excludes occurrences of the dataname where its value is defined, tested, used, set, or modified. This is the default value for the DATA name operand.
Use	Excludes occurrences of the dataname where its value is being tested or used.
Mod	Excludes occurrences of the dataname where its value is being set or modified.
Def	Excludes definitions of the dataname in the DATA DIVISION
ALias	Includes occurrences of the aliases for the dataname. These are the valid aliases: <ul style="list-style-type: none"> <li>• Parent - higher level group item</li> <li>• Child - lower level item</li> <li>• Rename/Redefinition - renamed, redefined, or 88 level items</li> </ul> This is the default value for the DATA name operand.
NOAlias	Ignores aliases for the specified dataname.
DIRect	Excludes occurrences of the dataname (and aliases if specified) where it is directly tested, used, set, modified, or defined (based on the REF, USE, MOD, or DEF selection). The default for the DATA name operand is direct; however, if you specify the SIZE, VALUE, or LEVELS operand, INDIRECT is assumed.
INDirect	Includes any dataname indirectly affected by the specified dataname (and aliases if specified). The indirect data item can be further qualified using the SIZE, VALUE, and LEVELS subordinate operands. These subordinate operands indicate the type of indirect reference to be located. SIZE and all levels are the defaults for the INDIRECT operand.
SIZE	Includes occurrences of datanames that could be directly or indirectly affected if the size of the specified dataname were to be changed. SIZE is only valid with the INDIRECT operand. This is the default for the INDIRECT operand.

Operand	Description
VALue	Includes occurrences of datanames that could either be directly or indirectly affected if the value of the specified dataname were to be changed. The LEVELS subordinate operand is not used with VALUE. VALUE is only valid with the INDIRECT operand.
LEVels <i>n</i>	Includes all of the data items affected within the specified number of indirect levels. The VALUE subordinate operand is not used with LEVELS. All levels is the default for the LEVELS operand. LEVELS is only valid with the INDIRECT operand.
PATTERN	Indicates that the characters that follow are part of a string. This is an optional keyword
<i>string</i>	Specifies a string of alphanumeric or DBCS characters. If the pattern string contains blanks, it must be enclosed in single or double quotes. You can further qualify the pattern string by using the WORD, PREFIX, or SUFFIX subordinate operands. These subordinate operands describe how the pattern string is to be used.
X' <i>string</i> '	Specifies the hexadecimal string option. Specific unprintable characters may be specified by giving the EBCDIC hexadecimal value. The string must be enclosed in single or double quotes.
T' <i>string</i> '	Specifies the text string option. You may enter a character string regardless of upper or lowercase by using the text option. The string must be enclosed in single or double quotes.
P' <i>string</i> '	Specifies the picture string option. A string profile may be entered instead of exact characters. The string must be enclosed in single or double quotes. Insight defines nine special characters for use in picture strings. These special characters can be combined with other characters. These are the special characters: P'=' Any character P'~' Any nonblank character P'!' Any nondisplay character P'#' Any numeric character P'-' Any character P'@' Any character P'<' Any character P'>' Any character P'\$' Any character

Operand	Description
WORD	Specifies the specified pattern string preceded and followed by any non-alphanumeric character (except a hyphen).
PREFIX	Specifies a word that begins with the specified pattern string.
SUFFIX	Specifies a word that ends with the specified pattern string.
Column1	Specifies the column number where the search is to begin.
Column2	Specifies the column number where the search is to end.
Blank	Excludes all lines from the display.
ALL	Excludes all lines from the display. The ALL operand cannot be entered with any other operands. This is the default.
Next	Searches forward from the cursor position to the next occurrence of the requested target.
Prev	Searches backward from the cursor position to the previous occurrence of the requested target.
First	Searches from the top of the source file to the first occurrence of the requested target.
LAST	Searches backward from the bottom of the source file to the first occurrence of the requested target.
All	Searches for all occurrences of the requested target and displays the number found in the short/long message field. This is the default value for the EXCLUDE command.
IN	Restricts the EXCLUDE command to the specified target type.

### Usage Notes

You can exclude lines by using the X (Exclude) and XX (Exclude Block) line commands on the View - Exclude Request pop-up. To display the View - Exclude Request pop-up, select View ► Exclude and press Enter. You can also exclude lines by using one of the Search pop-ups available on the Search pull-down.

The EXCLUDE command can be used to remove specific lines from the screen resulting from a previous command. Excluded lines are represented by a line of dashes and text stating n LINE(S) NOT DISPLAYED. The X or XX line commands can also be used to exclude lines from the screen.

To concatenate targets, place a plus sign (+) between the target names. For example, IO + CALL. These rules apply to concatenation:

- A concatenated dataname can be used wherever a dataname is valid. Dataname subordinate operands (REF, ALIAS, etc.) pertain to all datanames in a concatenated series.
- A concatenated set name can be used wherever a set name is valid. These sets can be concatenated:
  - LABEL
  - MARK
  - PERFRANGE
  - SUBSET
  - LINE
  - PROGRAM
  - PARAGRAPH
  - PATTERN
- The asterisk (\*) operand can be concatenated once with any number of other operands (e.g., \* + IO + MYSET); however, the \* operand cannot be concatenated to itself (\* + \* is invalid). The \* operand may appear in any order in the concatenated list.

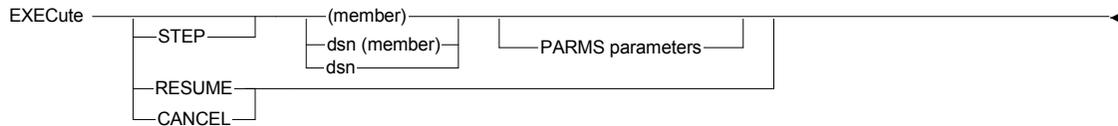
For COBOL II Release 3 or later programs, a data item, label, or program name that may be ambiguous or used multiple times can be qualified by using OF followed by the program name. For example, if the label P120-READ exists in another program and also in the program VIAIDEM1, then it can be qualified as P120-READ OF VIAIDEM1.

### **Example**

This command removes all unhighlighted lines from the screen:

```
EXCLUDE NHI
```

## EXECUTE Command



### Function

The EXECUTE command is used to read and execute the commands in a script file. Script files can contain EXECUTE commands that execute lower level (nested) script files. The STEP operand allows you to step through each script command.

### Operands

Operand	Description
STEP	Steps through the commands in the script file. Each command in the script file displays in the command input area. The displayed command can be changed or erased from the command input area. Press Enter to execute the displayed command. Processing of the script file continues in this manner until all commands have been displayed and/or executed.
RESUME	Executes the remaining script file commands without stepping through each. EXECUTE RESUME is entered after a script file is executed with the STEP operand specified.
CANCEL	Cancels the remaining script file commands. EXECUTE CANCEL can be entered after a script file is executed with the STEP operand specified.
<i>member</i>	Specifies the script member in a default script file defined during product installation. You can modify either the default script files or their concatenation sequence for your profile by using the Script file allocation action on the Options pull-down. The parentheses are required.
<i>dsn(member)</i>	Specifies the dataset and member name of a partitioned file that contains the script commands to be executed. This member must be in card image format (LRECL=80). The specified dataset member must be entered following TSO usage conventions in this format:  <code>'pds . dsn (member)'</code>  If you omit the quotes, your TSO prefix or user ID is the high-level qualifier.

Operand	Description
<i>dsm</i>	<p>Specifies the dataset name of a sequential file that contains the script commands to be executed. This dataset must be in card image format (LRECL=80). The specified dataset name must be entered following TSO usage conventions in this format:</p> <p style="text-align: center;"><i>'sequential.dataset.name'</i></p> <p>If the quotes are omitted, your TSO prefix or userID is the high-level qualifier.</p>
PARMS <i>parameters</i>	<p>Specifies parameters to be passed to the script being executed. Commands within a script file may contain substitution variables numbered &amp;1 through &amp;9. Parameter values are passed at execution time to the substitution variables using the PARMS operand of the EXECUTE command. Type the parameters values in the numerical order 1 through 9. If more than 9 parameter values are entered, the first nine are used and the remainder are ignored. If substitution variables are included in the script and parameters are not included in the EXECUTE command, the substitution variable values are passed as null.</p> <p>Substitution variables can be used anywhere in the executable script commands, but are not substituted if they occur in comment lines.</p> <p>Parameter values may contain any non-quoted literal without embedded blanks or any quoted literal. Quoted literals retain their quotes after substitution within the script file.</p>

### Usage Notes

The EXECUTE command reads the specified dataset and executes the script contained in the file. The script can contain EXECUTE commands that execute lower level (nested) script files. Script files that create a loop are recognized and an error message displays.

The STEP operand provides you a means of stepping through each script command. This allows you to modify or skip commands as desired.

Scripts must be sequential datasets or members of partitioned datasets. In both cases, scripts must be in card image format (LRECL=80). Scripts can be used to perform these repetitive tasks:

- Initiate a test session
- Set default values
- Set up a test session
- Execute a test
- Re-execute a test
- Execute a predefined command sequence

Comments can be included in script files by typing an asterisk (\*) in column one followed by the comment text.

Script files can also be executed from the File - Execute Script pop-up. Select File ▶ Execute to display the File - Execute Script pop-up.

## Examples

**Example 1.** Type this command to cause the first command in the script file to be displayed in the command input area:

```
EXECUTE STEP 'USERID.INS00015.VIASCRIP'
```

The command is executed and the next command in the script file displays in the command input area. You can change or bypass (erase) any command when it displays in the command input area.

**Example 2.** In this EXECUTE command, all script commands in member SCRIPT01 in the default script datasets are sequentially executed:

```
EXECUTE (SCRIPT01)
```

These commands can consist of any valid Insight primary command. For example, BRANCH, FX, LPRINT.

**Example 3.** In this example, the program name MYPROG and the dataname HIRE-DATE pass as parameters to the script in member SCRIPT02 in the default script files.

```
EXECUTE (SCRIPT02) PARMs MYPROG HIRE-DATE
```

If the script contains these commands, the parameters are substituted in the commands in place of the substitution variables to allow flexibility in using the script:

```
QUALIFY &1  
FX &2
```

These are the actual commands resulting from the parameter substitution:

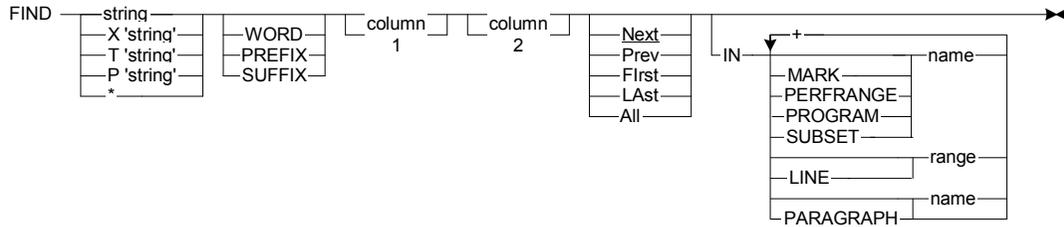
```
QUALIFY MYPROG  
FX HIRE-DATE
```

**Note:** \_\_\_\_\_

See ["PROCESS Command" on page 276](#) for additional uses of EXECUTE parameters.

---

## FIND Command



### Function

The FIND command searches for one or all occurrences of the specified character string. The syntax is similar to the ISPF/PDF FIND command. To locate DBCS strings, specify Double Byte Character Set (DBCS) identifiers or patterns with the FIND command.

### Operands

Operand	Description
<i>string</i>	Specifies a string of alphanumeric or DBCS characters. If the string contains blanks, it must be enclosed in single or double quotes. The string cannot contain mixed alphanumeric and DBCS characters. You can further qualify the string by using the WORD, PREFIX, or SUFFIX subordinate operands. These subordinate operands describe how the string is to be used:
X' <i>string</i> '	Specifies the hexadecimal string option. You may specify specific unprintable characters by giving the EBCDIC hexadecimal value. The string must be enclosed in single or double quotes.
T' <i>string</i> '	Specifies the text string option. You can enter a character string regardless of upper or lowercase by using the text option. The string must be enclosed in single or double quotes.
P' <i>string</i> '	Specifies the picture string option. You can enter a string profile instead of exact characters. The string must be enclosed in single or double quotes. Insight defines nine special characters for use in picture strings. These special characters can be combined with other characters:  P=' Any character P-' Any nonblank character

Operand	Description
P'.'	Any nondisplay character
P'#'	Any numeric character
P'-'	Any character
P'@'	Any character
P'<'	Any character
P'>'	Any character
P'\$'	Any character
asterisk (*)	Reuses the target of the previous FIND.
WORD	Specifies the string preceded and followed by any non-alphanumeric character (except a hyphen).
PREFIX	Specifies a word that begins with the specified string.
SUFFIX	Specifies a word that ends with the specified string.
Column1	Specifies the column number where the search is to begin.
Column2	Specifies the column number where the search is to end.
Next	Searches forward from the cursor position to the next occurrence of the requested target. This is the default value for the FIND command.
Prev	Searches backward from the cursor position to the previous occurrence of the requested target.
Ffirst	Searches from the top of the source file to the first occurrence of the requested target.
LAst	Searches backward from the bottom of the source file to the first occurrence of the requested target.
All	Searches for all occurrences of the requested target and displays the number found in the short/long message field.
IN	Restricts the FIND command to the specified target type.

Operand	Description								
MARK <i>name</i>	<p>Specifies a 1 to 10 alphanumeric character name given to a set or path using the COPY, MARK, MERGE, or RENAME commands, or one of these system-generated paths:</p> <table border="0"> <tr> <td>System generated path:</td> <td>Created by:</td> </tr> <tr> <td>TRACK   TRK</td> <td>A TRACE command.</td> </tr> <tr> <td>NETWORK   NET</td> <td>A FLOW command.</td> </tr> <tr> <td>SUBNET<sub>n</sub>   SUB<sub>n</sub></td> <td>A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (<i>n</i>th result).</td> </tr> </table>	System generated path:	Created by:	TRACK   TRK	A TRACE command.	NETWORK   NET	A FLOW command.	SUBNET <sub>n</sub>   SUB <sub>n</sub>	A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results ( <i>n</i> th result).
System generated path:	Created by:								
TRACK   TRK	A TRACE command.								
NETWORK   NET	A FLOW command.								
SUBNET <sub>n</sub>   SUB <sub>n</sub>	A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results ( <i>n</i> th result).								
<p><b>Note:</b></p> <p>The TRACK, NETWORK, and SUBNET paths can also be created using the Logic Order Search pop-ups (see <a href="#">"Logic Pull-down" on page 101</a>).</p>									
PERFRANGE <i>name</i>	Specifies the name in a PERFORM statement including all statements executed as a result of a PERFORM statement, or the name of any section contained in the Declaratives.								
PROGRAM <i>name</i>	Specifies the name of the main program or of any nested program representing all of the code contained in the program. This includes all of the programs physically nested inside the specified program.								
SUBSET <i>name</i>	Specifies a predefined COBOL language subset. See the <i>ASG-Insight User's Guide</i> for a list of each subset and its description.								
LINE <i>range</i>	Specifies a single line number or a range of lines. Line numbers display in columns 1 through 6 of Source View.								
PARAGRAPH <i>name</i>	Specifies any PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. The entire paragraph or section is included.								

### Usage Notes

You can also conduct a search by selecting a Search pop-up on the Search pull-down.

The search begins from the current line when the you specify the NEXT or PREV operands. When you specify the ALL operand, all lines are searched regardless of the current line or direction. Any excluded lines containing FIND targets are displayed and highlighted. The IN operand restricts the FIND command to only the targets you specified.

To concatenate targets, place a plus sign (+) between the target names. For example, IO + CALL. These are the rules for concatenation:

- A concatenated dataname wherever a dataname is valid. Dataname subordinate operands (REF, ALIAS, etc.) pertain to all datanames in a concatenated series.
- A concatenated set name can be used wherever a set name is valid. These sets can be concatenated:
  - LINE
  - PERFRANGE
  - PROGRAM
  - MARK
  - PARAGRAPH
  - SUBSET

For COBOL II Release 3 or later programs, you may qualify a data item, label, or program name that may be ambiguous or used multiple times by using OF followed by the program name. For example, if the label P120-READ exists in another program and also in the program VIAIDEM1, then you can qualify it as P120-READ OF VIAIDEM1.

### Examples

**Example 1.** In this command, all occurrences of XYZ are highlighted:

```
FIND 'XYZ' ALL
```

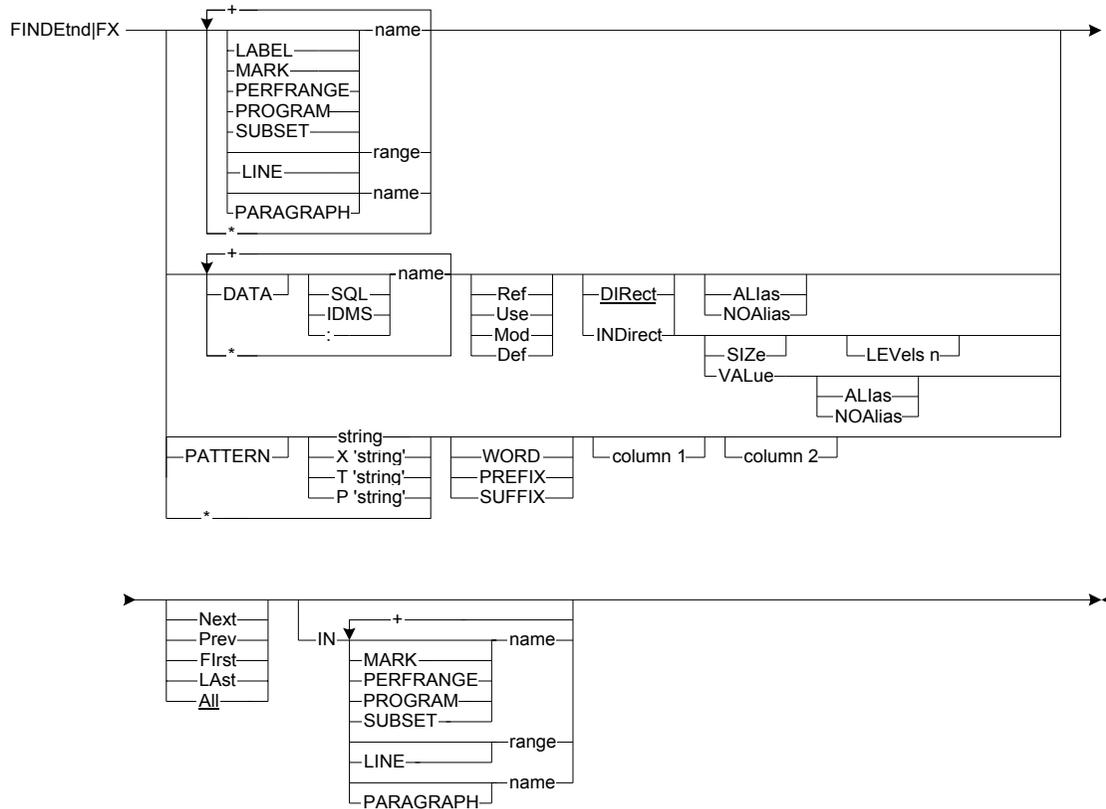
**Example 2.** In this command, the FIND command searches for the DATA NAME string without regard to case. The DATA NAME string can be uppercase, lowercase, or mixed case:

```
FIND T'DATA NAME'
```

**Example 3.** In this example, the FIND command searches for two non-blank characters separated by a blank:

```
FIND P'  -  -'
```

## FINDXTND Command



### Function

The FINDXTND command performs a COBOL intelligent search of the source code for either one or all occurrences of the specified target.

### Operands

Operand	Description
LABEL <i>name</i>	Specifies a PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. All transfers of control to the label name are included.

Operand	Description
MARK <i>name</i>	<p>Specifies a 1 to 10 alphanumeric character name given to a set or path using the COPY, MARK, MERGE, or RENAME commands, or one of these system-generated paths:</p> <p>System generated path: Created by:</p> <p>TRACK   TRK      A TRACE command.</p> <p>NETWORK   NET    A FLOW command.</p> <p>SUBNET<sub><i>n</i></sub>   SUB<sub><i>n</i></sub>    A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (<i>n</i><sup>th</sup> result).</p> <p><b>Note:</b></p> <p>The TRACK, NETWORK, and SUBNET paths can also be created using the Logic Order Search pop-ups (see <a href="#">"Logic Pull-down" on page 101</a>).</p>
PERFRANGE <i>name</i>	<p>Specifies the name specified in a PERFORM statement including all statements executed as a result of a PERFORM statement, or the name of any section contained in the Declaratives.</p> <p>This operand is valid in Source View only</p>
PROGRAM <i>name</i>	<p>Specifies the name of the main program or any nested program representing all the code contained in the program. This includes all the programs physically nested inside the specified program.</p>
SUBSET <i>name</i>	<p>Specifies a predefined COBOL language subset. See the <i>ASG-Insight User's Guide</i> for a list of each subset and its description.</p>
LINE <i>range</i>	<p>Specifies a single line number or range of lines. Line numbers display in columns 1 through 6 of Source View.</p>
PARAGRAPH <i>name</i>	<p>Specifies a PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. The entire paragraph or section is included.</p>
asterisk (*)	<p>Reuses the target of the previous search, exclude, find, highlight, or scroll action or command.</p>
DATA <i>name</i>	<p>Specifies a COBOL dataname or qualified COBOL dataname. Dataname refers to any valid COBOL reference for a data element. These subordinate operands apply to the dataname and can be used to further qualify the EXCLUDE command: REF, USE, MOD, DEF, ALIAS, NOALIAS, DIRECT, and INDIRECT. The defaults are REF, ALIAS, and DIRECT.</p>

Operand	Description
SQL <i>name</i>	Excludes datanames that are DB2/SQL variables only.
IDMS <i>name</i>	Excludes datanames that are IDMS variables only.
: <i>name</i>	Excludes datanames that are COBOL variables only.
Ref	Excludes occurrences of the dataname where its value is defined, tested, used, set, or modified. This is the default value for the DATA name operand.
Use	Excludes occurrences of the dataname where its value is being tested or used.
Mod	Excludes occurrences of the dataname where its value is being set or modified.
Def	Excludes definitions of the dataname in the DATA DIVISION
ALias	Includes occurrences of the aliases for the dataname. These are the valid aliases: <ul style="list-style-type: none"> <li>• Parent - higher level group item</li> <li>• Child - lower level item</li> <li>• Rename/Redefinition - renamed, redefined, or 88 level items</li> </ul> This is the default value for the DATA name operand.
NOAlias	Ignores aliases for the specified dataname.
DIRect	Excludes occurrences of the dataname (and aliases if they are specified) where it is directly tested, used, set, modified, or defined (based on the REF, USE, MOD, or DEF selection). The default for the DATA name operand is direct; however, if the SIZE, VALUE, or LEVELS operand is specified, INDIRECT is assumed.
INDirect	Includes any dataname indirectly affected by the specified dataname (and aliases if specified). The indirect data item can be further qualified using the SIZE, VALUE, and LEVELS subordinate operands. These subordinate operands indicate the type of indirect reference to be located. SIZE and all levels are the defaults for the INDIRECT operand.
SIZE	Includes occurrences of datanames that could be directly or indirectly affected if the size of the specified dataname were to be changed. SIZE is only valid with the INDIRECT operand. This is the default for the INDIRECT operand.

Operand	Description
VALue	Includes occurrences of datanames that could be directly or indirectly affected if the value of the specified dataname were to be changed. The LEVELS subordinate operand is not used with VALUE. VALUE is only valid with the INDIRECT operand.
LEVelS <i>n</i>	Includes all data items affected within the specified number of indirect levels. The VALUE subordinate operand is not used with LEVELS. All levels is the default for the LEVELS operand. LEVELS is only valid with the INDIRECT operand.
PATTERN	Indicates the characters that follow are part of a string.
<i>string</i>	Specifies a string of alphanumeric or DBCS characters. If the pattern string contains blanks, it must be enclosed in single or double quotes. You can further qualify the pattern string by using the WORD, PREFIX, or SUFFIX subordinate operands. These subordinate operands describe how the pattern string is to be used.
X' <i>string</i> '	Specifies the hexadecimal string option. You can specify specific unprintable characters by giving the EBCDIC hexadecimal value. The string must be enclosed in single or double quotes.
T' <i>string</i> '	Specifies the text string option. Use the text option to enter a character string regardless of upper or lowercase. The string must be enclosed in single or double quotes.
P' <i>string</i> '	Specifies the picture string option. You may enter a string profile instead of exact characters. The string must be enclosed in single or double quotes. Insight defines nine special characters for use in picture strings. These special characters can be combined with other characters. These are the special characters and their meanings: P'=' Any character P'~' Any nonblank character P'!' Any nondisplay character P'#' Any numeric character P'-' Any character P'@' Any character P'<' Any character P'>' Any character P'\$' Any character
WORD	Specifies the pattern string preceded and followed by any non-alphanumeric character (except a hyphen).

Operand	Description
PREFIX	Specifies a word that begins with the specified pattern string.
SUFFIX	Specifies a word that ends with the specified pattern string.
Column1	Specifies the column number where the search is to begin.
Column2	Specifies the column number where the search is to end.
ALL	Excludes all lines from the display. You cannot enter the ALL operand with any other operands. This is the default.
Next	Searches forward from the cursor position to the next occurrence of the requested target.
Prev	Searches backward from the cursor position to the previous occurrence of the requested target.
Ffirst	Searches from the top of the source file to the first occurrence of the requested target.
LAst	Searches backward from the bottom of the source file to the first occurrence of the requested target.
All	Searches for all occurrences of the requested target and displays the number found in the short/long message field. This is the default value for the EXCLUDE command.
IN	Restricts the EXCLUDE command to the specified target type.

### **Usage Notes**

If you are using ISPF 4.1 or later, make sure the Long Message in pop-up ISPF setting is disabled when you execute the FX command. If this setting is not disabled, a portion of the display screen is overlaid with the FX results formatted in a long message pop-up. Additional problems can result when you execute the RFIND command following an FX command.

You can also conduct a search by selecting a Search pop-up on the Search pull-down. See the online help for more information.

To concatenate targets, place a plus sign (+) between the target names. For example, IO+CALL. These rules apply to concatenation:

- A concatenated dataname can be used wherever a dataname is valid. Dataname subordinate operands (REF, ALIAS, etc.) pertain to all datanames in a concatenated series.
- A concatenated set name can be used wherever a set name is valid. These sets can be concatenated:
  - LABEL
  - MARK
  - PERFRANGE
  - SUBSET
  - LINE
  - PROGRAM
  - PARAGRAPH
  - PATTERN
- The asterisk (\*) operand can be concatenated once with any number of other operands (e.g., \* + IO + MYSET); however, the \* operand cannot be concatenated to itself (\* + \* is invalid). The \* operand may appear in any order in the concatenated list.

For COBOL II Release 3 or later programs, you can qualify a data item, label, or program name that may be ambiguous or used multiple times by using OF followed by the program name. For example, if the label P120-READ exists in another program and also in the program VIAIDEM1, then it can be qualified as P120-READ OF VIAIDEM1.

Highlighting is used to indicate the occurrences found. Lines already highlighted are reset, so that only the results of the current command are highlighted. If you exclude lines containing targets, they redisplay on the screen. Tags are placed on the source code lines indicating the type of target found. These tags are placed in columns 73 through 80:

Tag	Description
LABEL <i>name</i>	LABELREF
MARK <i>name</i>	TRACK NETWORK MARK SUBNET
PERFRANGE <i>name</i>	PERF RNG
PROGRAM <i>name</i>	PROGRAM PGM tag

Tag	Description
SUBSET <i>name</i>	Specifies a predefined COBOL subset name. These are the valid subsets: ASsignment    DEBUg    IO CALL    DEFinition    LABEL Cics    DIRective    MAINline COBOLII    DIVision    MATH COBOL/370    DL/I   DL/1    Output COMment    DML    PARagraph CONditional    ENtry    PERform COPy    EXIt   PGM    ExitRETurn DB2/SQL    FALLthrough    SECTion DDL    GOto    SORTMerge DEAD    IDMS    STRucture DEADCode    INCLude    TESTed DEADData    Input    UNTested
SUBSET <i>name</i> - SCREEN	HIGH NONHIGH EXCLUDE NONX
SUBSET <i>name</i> - TRACE tags	DECISION OPTION START TARGET
LINE <i>range</i>	LINE RNG
PARAGRAPH <i>name</i>	PARAGRAPH
DATA <i>name</i>	DATA REF DATA USE DATA MOD DATA DEF

Tag	Description
PATTERN <i>string</i>	<p>Specifies the pattern string. The NEXT and PREV operands start the search from the current position and locate the closest occurrence. If you specify ALL, all lines are searched regardless of the current line or direction. A message displays indicating the number of targets found.</p> <p>When you enter an unqualified dataname and more than one occurrence of the specified dataname exists, all occurrences are found. This allows you to see all like datanames but under different group level names, then allows you to select the appropriate dataname. A message displays if you enter an invalid qualification.</p> <p>When you enter a dataname with the REF operand, the screen is positioned to show the first occurrence of the dataname in the PROCEDURE DIVISION. DEF information in the DATA DIVISION is also highlighted.</p>
FX INDIRECT	<p>Locates the specified indirect dataname. A dataname can impact other datanames in source statements. These datanames can be subsequently propagated indirectly through other statements, and referenced at each level. For example, consider this code:</p> <pre data-bbox="716 1081 907 1173">MOVE A TO B MOVE B TO C MOVE C TO D</pre> <p>When the dataname of interest is A, the first statement contains a direct reference to A. The direct reference to a dataname is level 1. The last two statements contain indirect references to A, based on the propagation of the dataname A by moving its value to B, then to C, then to D. The second statement is a level 2 reference to dataname A, and the last statement is a level 3 reference to dataname A.</p> <p>All levels of the indirect impact to a dataname display, or you can view one level at a time using the LEVELS operand. For example, to locate all datanames that could be indirectly affected by a change in the size of the specified dataname, use the command FX A SIZE. Because the LEVELS operand is not specified, the default of ALL LEVELS is used. This results in a complete list of all datanames that must be reviewed (for a size change of the specified dataname). If the LEVELS operand is specified (e.g., FX A SIZE LEVELS 2), only the results of that many levels are highlighted. The ripple effect can be more clearly identified by starting with LEVELS 1, then increasing the number of levels to be searched.</p>

Tag	Description
	<p>When using INDIRECT SIZE, any occurrence of an affected dataname in the USING clause of a CALL statement is tagged as a MOD. The CALLED subprogram must be examined for other datanames affected as a result of the CALL. IN target can be used to restrict the search to the specified IN target.</p> <p>When the REF operand is specified or used by default, all appropriate references are highlighted. If the reference is a DEFINITION, USE, or MODIFICATION, the line is tagged accordingly. If the line contains a USE and a MODIFICATION, it is tagged as a REFERENCE.</p>
DBCS <i>strings</i>	The FINDXTND command supports DBCS strings. See <a href="#">"FIND Command" on page 212</a> for detailed information.

## Examples

**Example 1.** This FINDXTND command searches backward for the previous occurrence of a use of DATE-CODE or its aliases:

```
FINDXTND DATE-CODE USE PREV
```

**Example 2.** In this command, the definition of DATE-CODE (without any aliasing) in the DATA DIVISION is highlighted:

```
FX DATE-CODE DEF NOALIAS
```

**Example 3.** In this command, all occurrences of the SWITCH field used in conditional statements are highlighted:

```
FX SWITCH IN COND
```

**Example 4.**

```
01 A.
   05 B          PIC X.
   05 C.
       10 D      PIC X.
       10 E      PIC X.
01 Z REDEFINES A.
   05 Z1         PIC XXX.
66 L RENAMES Z.
```

These statements apply if you enter a FINDXTND command for C with ALIAS specified:

- A is the parent.
- D and E are the children.
- Z is a redefine/rename.
- Z1 is a redefine/rename.
- L is a rename.

#### Example 5.

```
000010 READ INFILE INTO A.
000020 MOVE A TO B.
000030 MOVE B TO C.
000040 WRITE OUTFILE FROM C.
```

The MOD operand is used to locate only those places where B is directly or indirectly set or modified. The fields in lines 10 and 20 are highlighted since B is modified by A on line 20, and A is modified by the READ statement on line 10.

This FINDXTND command shows the possible origin of the data value:

```
FX B MOD INDIRECT VALUE
```

The USE operand is used to locate every place B is directly or indirectly used. The fields in lines 30 and 40 are highlighted because both statements use B. The FINDXTND command below shows that the value of B is used to modify C on line 30. On line 40, the value of C that was received from B is used. This FINDXTND command shows all possible destinations of a data field:

```
FX B USE INDIRECT VALUE
```

#### Example 6.

```
000010 01 A.
000020      05 B.      PIC XX.
000070 01 C.
000080      05 D      PIC XX.
001010      MOVE B TO X.
001020      MOVE C TO A.
001030      MOVE SPACES TO D.
```

The FINDXTND command below is used to locate all groups whose definitions might have to be changed every time the definition for B changes. The results are lines 10, 20, 70, 80, 1010, and 1020. Any change in the size of B is a change in the size of A as well. This means the definition for C should be changed since A now has a different record layout, and the statement on line 1020 would not execute as intended. D is also shown as a result because it is at the same offset within C as B is within A, and is the same size as B.

```
FX B REF INDIRECT
```

**Note:** \_\_\_\_\_

The number of results from FX INDIRECT REF depends greatly on the target name. For best results, select a target that is the smallest group item whose subfields are to be modified.

\_\_\_\_\_

**Example 7.** These examples use this code to illustrate the LEVELS, VALUE, and SIZE operands:

```
000010 MOVE A TO B.  
000020 MOVE X TO B.  
000080 MOVE Y TO X.  
000100 MOVE Y TO A.
```

This command identifies the datanames in the program directly affected by a change in A. This list consists of only dataname A and its related aliases.

```
FX A INDIRECT LEVELS 0
```

This command identifies the statement (move of size from A to B) that shows B might be affected by this move. This statement (line 10) and the definitions of A and B are highlighted.

```
FX A INDIRECT LEVELS 1
```

This command highlights the definitions of A, B, and X, plus the statements on lines 10 and 20.

```
FX A INDIRECT LEVELS 2
```

This command identifies all direct and indirect uses of the value of A. The statement on line 10 would be highlighted.

```
FX A USE INDIRECT VALUE
```

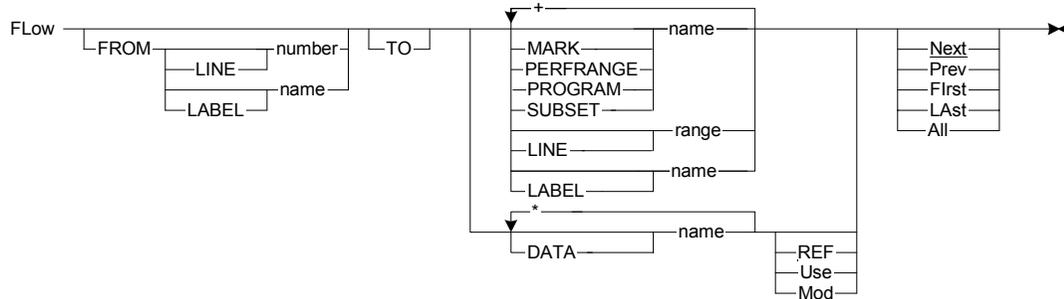
This command identifies all direct and indirect modifications to the value of A. The statement on line 100 would be highlighted.

```
FX A MOD INDIRECT VALUE
```

This command identifies where A is affected by a change in the size of the target dataname. The statements on line 20 and 100 would be highlighted.

```
FX A USE INDIRECT SIZE
```

## FLOW Command



### Function

The FLOW command follows all possible execution paths from a given point in a program searching for the specified target(s). The identified paths are stored with mark names of NETWORK and SUBNET $n$ .

### Operands

Operand	Description								
FROM	Specifies an optional keyword that is entered with a line number or label to indicate where the FLOW command is to begin.								
LINE <i>number</i>	Specifies the line number where the FLOW command is to begin.								
LABEL <i>name</i>	Specifies the paragraph or section name of the PROCEDURE DIVISION or the literals PROCEDURE or PROC where the FLOW command is to begin.								
TO	Specifies an optional keyword that is followed by a target for the FLOW command.								
MARK <i>name</i>	Specifies a 1 to 10 alphanumeric character name given to a set or path using the COPY, MARK, MERGE, or RENAME commands, or one of these system-generated paths: <table border="0" style="margin-left: 20px;"> <tr> <td>System generated path:</td> <td>Created by:</td> </tr> <tr> <td>TRACK   TRK</td> <td>A TRACE command.</td> </tr> <tr> <td>NETWORK   NET</td> <td>A FLOW command.</td> </tr> <tr> <td>SUBNET<math>n</math>   SUB<math>n</math></td> <td>A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (<math>n</math>th result).</td> </tr> </table>	System generated path:	Created by:	TRACK   TRK	A TRACE command.	NETWORK   NET	A FLOW command.	SUBNET $n$   SUB $n$	A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results ( $n$ th result).
System generated path:	Created by:								
TRACK   TRK	A TRACE command.								
NETWORK   NET	A FLOW command.								
SUBNET $n$   SUB $n$	A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results ( $n$ th result).								

Operand	Description																																							
	<p><b>Note:</b></p> <p>The TRACK, NETWORK, and SUBNET paths can also be created using the Logic Order Search pop-ups (see <a href="#">"Logic Pull-down" on page 101</a>).</p>																																							
PERFRANGE <i>name</i>	<p>Specifies the name in a PERFORM statement including all statements executed as a result of a PERFORM statement, or the name of any section contained in the Declaratives.</p> <p>This operand is valid in Source View only</p>																																							
PROGRAM <i>name</i>	<p>Specifies the name of the main program or any nested program representing all the code contained in the program. This includes all the programs physically nested inside the specified program.</p>																																							
SUBSET <i>name</i>	<p>Specifies a predefined subset. These are the valid subsets:</p> <table border="0"> <tr> <td>ASsignment</td> <td>DEBUg</td> <td>IO</td> </tr> <tr> <td>CALL</td> <td>DEFinition</td> <td>LABEL</td> </tr> <tr> <td>CIcs</td> <td>DIRective</td> <td>MAINline</td> </tr> <tr> <td>COBOLII</td> <td>DIVision</td> <td>MATH</td> </tr> <tr> <td>COBOL/370</td> <td>DL/I   DL/1</td> <td>Output</td> </tr> <tr> <td>COMment</td> <td>DML</td> <td>PARagraph</td> </tr> <tr> <td>CONditional</td> <td>ENtry</td> <td>PERform</td> </tr> <tr> <td>COPY</td> <td>EXIt   PGM</td> <td>ExitRETurn</td> </tr> <tr> <td>DB2/SQL</td> <td>FALLthrough</td> <td>SECTion</td> </tr> <tr> <td>DDL</td> <td>GOto</td> <td>SORTMerge</td> </tr> <tr> <td>DEAD</td> <td>IDMS</td> <td>STructure</td> </tr> <tr> <td>DEADCode</td> <td>INClude</td> <td>TESTed</td> </tr> <tr> <td>DEADData</td> <td>Input</td> <td>UNTested</td> </tr> </table> <p>A screen subset:</p> <p>Highlighted   HI  NONHighlighted   NHI  Excluded   X  NONExcluded   NX</p>	ASsignment	DEBUg	IO	CALL	DEFinition	LABEL	CIcs	DIRective	MAINline	COBOLII	DIVision	MATH	COBOL/370	DL/I   DL/1	Output	COMment	DML	PARagraph	CONditional	ENtry	PERform	COPY	EXIt   PGM	ExitRETurn	DB2/SQL	FALLthrough	SECTion	DDL	GOto	SORTMerge	DEAD	IDMS	STructure	DEADCode	INClude	TESTed	DEADData	Input	UNTested
ASsignment	DEBUg	IO																																						
CALL	DEFinition	LABEL																																						
CIcs	DIRective	MAINline																																						
COBOLII	DIVision	MATH																																						
COBOL/370	DL/I   DL/1	Output																																						
COMment	DML	PARagraph																																						
CONditional	ENtry	PERform																																						
COPY	EXIt   PGM	ExitRETurn																																						
DB2/SQL	FALLthrough	SECTion																																						
DDL	GOto	SORTMerge																																						
DEAD	IDMS	STructure																																						
DEADCode	INClude	TESTed																																						
DEADData	Input	UNTested																																						

Operand	Description
	Tagged lines subsets of tags in columns 73 through 80.  <b>Note:</b> _____ See the <i>ASG-Insight User's Guide</i> for a description of each subset _____
LINE <i>range</i>	Specifies a single line number or range of lines. Line numbers display in columns 1 through 6 of Source View.
LABEL <i>name</i>	Specifies any PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. All transfers of control to the label name are included.
DATA <i>name</i>	Specifies a COBOL dataname or qualified COBOL dataname. Dataname refers to any valid COBOL reference for a data element. These subordinate operands apply to the dataname and can be used to further qualify the EXCLUDE command: REF, USE, MOD, DEF, ALIAS, NOALIAS, DIRECT, and INDIRECT. The defaults are REF, ALIAS, and DIRECT.
Ref	Excludes occurrences of the dataname where its value is defined, tested, used, set, or modified. This is the default value for the DATA name operand.
Use	Excludes occurrences of the dataname where its value is being tested or used.
Mod	Excludes occurrences of the dataname where its value is being set or modified.
Next	Searches forward from the cursor position to the next occurrence of the requested target.
Prev	Searches backward from the cursor position to the previous occurrence of the requested target.
Forward	Flows forward from the current cursor position to all occurrences.
Backward	Flows backward from the current cursor position to all occurrences.

### Usage Notes

You can also identify execution paths in a program by using the Logic Order Search pop-ups that can create paths with mark names of NETWORK and SUBNET $n$ . See [Chapter 5, "Logic," on page 99](#) for more information about the Logic Order Search pop-ups.

The FLOW command is used to determine the execution flow of a program and indicates if the specified targets can be reached from a given point. A path of the executable statements that are followed is captured and called NETWORK. You can save the NETWORK and use it for additional analysis. Display a NETWORK by using the FINDXTND NET command.

To concatenate targets, place a plus sign (+) between the target names. For example, IO+CALL. These rules apply to concatenation:

- A concatenated dataname can be used wherever a dataname is valid. Dataname subordinate operands (REF, USE, MOD) pertain to all datanames in a concatenated series.
- A concatenated set name can be used wherever a set name is valid. These sets can be concatenated:
  - LABEL
  - MARK
  - PERFRANGE
  - LINE
  - PROGRAM
  - SUBSET

For COBOL II Release 3 or later programs, you can qualify a data item, label, or program name that may be ambiguous or used multiple times by using OF followed by the program name. For example, if the label P120-READ exists in another program and also in the program VIAIDEM1, then it can be qualified as P120-READ OF VIAIDEM1.

The FROM operand is used as the starting point for the FLOW command. If you do not specify a FROM operand, the cursor position is used as the starting point.

If you position the cursor on an executable COBOL statement, the starting point of the search is based on the location of the cursor as well as the target and direction specified in the FLOW command. For example:

```
FLOW data-name forward
```

If the cursor is before the statement, the statement is included.

```
FLOW data-name backward
```

If you place the cursor is after the last character of the statement, the statement is included.

If the starting point is an inexecutable COBOL statement, these general rules apply:

- Comment lines and compiler directives occurring at the end of an executable code block are considered part of the paragraph or section that follows. The search begins at the paragraph or section name that follows.

- A line followed by one or more executable statements prior to the succeeding section, or a line that is part of the last section, is considered part of the preceding section and the search begins at the end of the statement, before the comment or directive.
- Blank lines that occur at the end of an executable code block, and that are not preceded by a comment or compiler directive, are considered part of the preceding paragraph or section. The search begins at the end of the last executable statement in the preceding paragraph.
- A blank line that follows all executable statements of the preceding paragraph, and that is preceded by a comment or compiler directive, is considered part of this paragraph. The search begins at the paragraph or section name in this paragraph.
- A blank line that is followed by one or more executable statements, or that is part of the last paragraph or section of the program, is considered part of the preceding paragraph. The search begins at the end of the preceding paragraph.

### Special Conditions

If you specify a dataname with USE NEXT or USE PREV, special conditions are considered. When the FLOW command searches for the next use, the value currently in the dataname is the target of the search. If a modification of that variable is encountered along a path, that portion of the execution path is not reflected in the result. The same is true for PREV. FLOW locates all uses occurring before a modification of the dataname. For example:

```
MOVE ZIP-CODE TO OUT-ZIP-CODE    (ZIP-CODE = 46201)
.
MOVE HOLD-ZIP TO ZIP-CODE        (HOLD-ZIP = 93721)
.
MOVE ZIP-CODE TO OUT-ZIP-CODE    (ZIP-CODE = 93721)
```

If you enter the FLOW USE PREV command on the last line, the first line would not be shown as a result because the value of ZIP-CODE is modified between the starting and ending points.

### Example

The FLOW command below flows to the previous uses of LOAN-AMT where the same value was used.

```
FLOW LOAN-AMT USE PREV
```

To show the network produced by the FLOW command, type this command:

```
FX NETWORK
```

## GOBACK Command



### Function

The GOBACK command is used to reverse a CALL or VIEW command and return to a Source View session.

### Operands

Operand	Description
Blank	Specifies that the program where the last CALL or VIEW was executed is returned to the Source View. Marks and equates are saved or not saved, depending on the values specified on the Options - Product Parameters pop-up.
<i>pgm</i>	Specifies a program or entry point in the AKR. A Source View session must currently be active for this program.
Save	Saves marks and equates created for this program in the AKR. This operand overrides the values on the Options - Product Parameters pop-up.
Nosave	Specifies that marks and equates created for this program are not saved. This operand overrides the values on the Options - Product Parameters pop-up.

### Usage Notes

The GOBACK command ends the current Source View session and returns to the Source View session where a CALL or VIEW command was entered. Specifying a program name returns to a particular program in the calling chain.

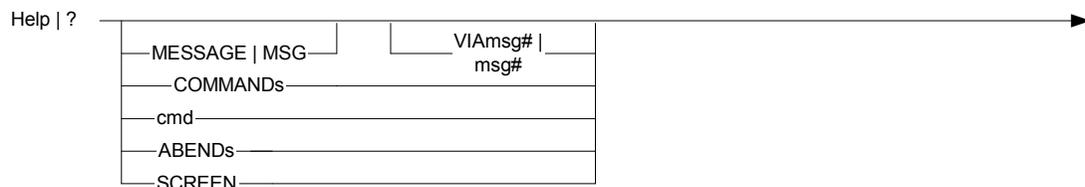
### Example

The current program is VIAIDEMO. To display the VIAIDEM1 program on the Source View screen, type

```
CALL VIAIDEM1
```

After the above command finishes executing, type GOBACK to return to the VIAIDEMO Source View session.

## HELP Command



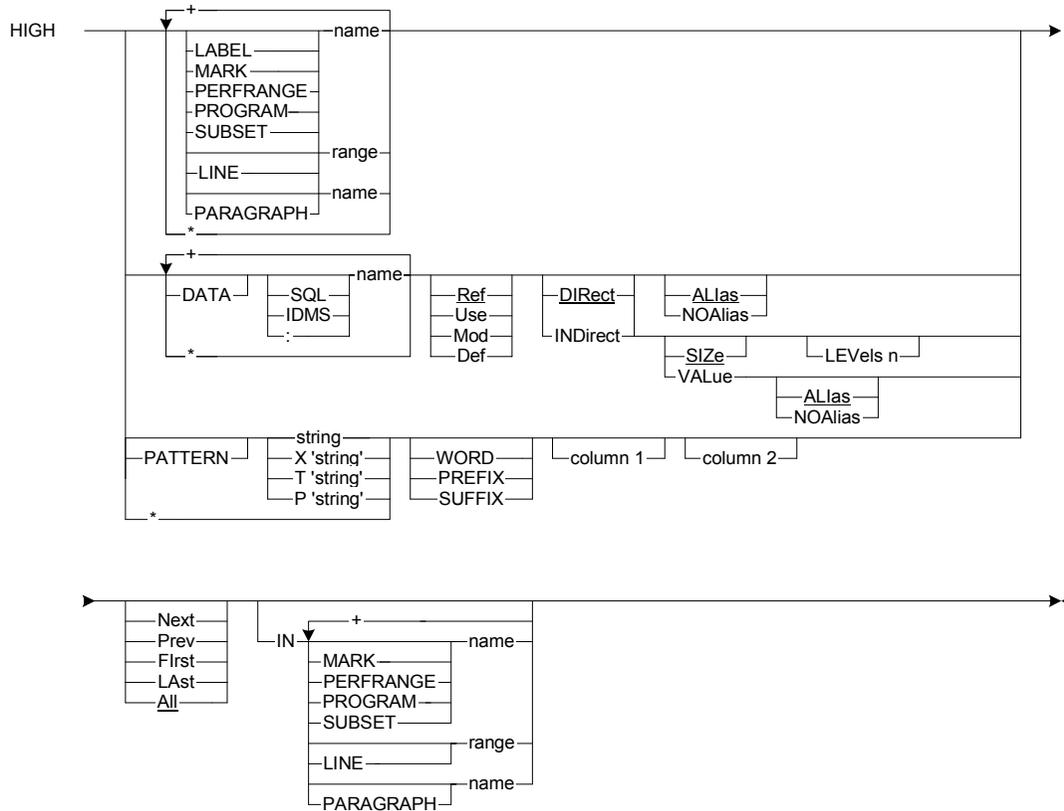
### Function

The HELP command displays information about the current Insight screen, pop-up, commands, messages, or abend codes.

### Operands

Operand	Description
Blank	Displays the Help Tutorial for the current screen or pop-up, that describes all the fields on the screen or pop-up, and any special processing considerations. If a message displays, type HELP with no operands to display the Help Explanation and Action Panel for the current message.
MESSAGE   MSG	Displays the Help Explanation and Action Panel, that shows the specified short and long message, an explanation of the current message, and any actions to be performed.
ASGmsg#   msg#	Specifies the message number where the Help Explanation and Action Panel screen is to be displayed. This number consists of 1 to 4 digits. It is not necessary to enter leading zeros.
COMMANDs	Displays a list of all Insight primary commands. Information for a particular command can then be displayed by selecting the appropriate number.
cmd	Specifies an Insight primary command. When help is requested for a command, a long message displays giving a brief description of that command. Requesting help again then displays more detailed information about the command.
ABENDs	Displays the Abends screen that lists the types of abend codes.
SCREEN	Displays help for the current screen or pop-up, that describes all fields on the screen or pop-up, and any special processing considerations.

## HIGH Command



### Function

The HIGH command highlights source code lines containing the specified targets. Lines already highlighted are not reset.

### Operands

Operand	Description
LABEL <i>name</i>	Specifies the paragraph or section name of the PROCEDURE DIVISION or the literals PROCEDURE or PROC where the FLOW command is to begin.
MARK <i>name</i>	Specifies a 1 to 10 alphanumeric character name given to a set or path using the COPY, MARK, MERGE, or RENAME commands, or one of these system-generated paths:

Operand	Description																																							
	<p>System generated path: Created by:</p> <p>TRACK   TRK A TRACE command.</p> <p>NETWORK   NET A FLOW command.</p> <p>SUBNET<sub>n</sub>   SUB<sub>n</sub> A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (n<sup>th</sup> result).</p>																																							
	<p><b>Note:</b></p> <p>The TRACK, NETWORK, and SUBNET paths can also be created using the Logic Order Search pop-ups (see <a href="#">"Logic Pull-down" on page 101</a>).</p>																																							
PERFRANGE <i>name</i>	<p>Specifies the name in a PERFORM statement including all of the statements executed as a result of a PERFORM statement, or the name of any section contained in the Declaratives.</p> <p>This operand is valid in Source View only</p>																																							
PROGRAM <i>name</i>	<p>Specifies the name of the main program or of any nested program representing all of the code contained in the program. This includes all of the programs physically nested inside the specified program.</p>																																							
SUBSET <i>name</i>	<p>Specifies a predefined subset name. These are the valid subsets:</p> <table border="0"> <tr> <td>ASsignment</td> <td>DEBUg</td> <td>IO</td> </tr> <tr> <td>CAll</td> <td>DEFinition</td> <td>LABel</td> </tr> <tr> <td>CIcs</td> <td>DIRective</td> <td>MAINline</td> </tr> <tr> <td>COBOLII</td> <td>DIVision</td> <td>MATH</td> </tr> <tr> <td>COBOL/370</td> <td>DL/I   DL/1</td> <td>Output</td> </tr> <tr> <td>COMment</td> <td>DML</td> <td>PARagraph</td> </tr> <tr> <td>CONditional</td> <td>ENtry</td> <td>PERform</td> </tr> <tr> <td>COPy</td> <td>EXIt   PGM</td> <td>ExitRETurn</td> </tr> <tr> <td>DB2/SQL</td> <td>FALLthrough</td> <td>SECTion</td> </tr> <tr> <td>DDL</td> <td>GOto</td> <td>SORTMerge</td> </tr> <tr> <td>DEAD</td> <td>IDMS</td> <td>STructure</td> </tr> <tr> <td>DEADCode</td> <td>INClude</td> <td>TESTed</td> </tr> <tr> <td>DEADData</td> <td>Input</td> <td>UNTested</td> </tr> </table>	ASsignment	DEBUg	IO	CAll	DEFinition	LABel	CIcs	DIRective	MAINline	COBOLII	DIVision	MATH	COBOL/370	DL/I   DL/1	Output	COMment	DML	PARagraph	CONditional	ENtry	PERform	COPy	EXIt   PGM	ExitRETurn	DB2/SQL	FALLthrough	SECTion	DDL	GOto	SORTMerge	DEAD	IDMS	STructure	DEADCode	INClude	TESTed	DEADData	Input	UNTested
ASsignment	DEBUg	IO																																						
CAll	DEFinition	LABel																																						
CIcs	DIRective	MAINline																																						
COBOLII	DIVision	MATH																																						
COBOL/370	DL/I   DL/1	Output																																						
COMment	DML	PARagraph																																						
CONditional	ENtry	PERform																																						
COPy	EXIt   PGM	ExitRETurn																																						
DB2/SQL	FALLthrough	SECTion																																						
DDL	GOto	SORTMerge																																						
DEAD	IDMS	STructure																																						
DEADCode	INClude	TESTed																																						
DEADData	Input	UNTested																																						

Operand	Description
	<p>A screen subset:</p> <p>Highlighted   HI  NONHighlighted   NHI  Excluded   X  NONExcluded   NX</p> <p>Tagged lines subsets of tags in columns 73 through 80.</p> <p><b>Note:</b> _____  See the <i>ASG-Insight User's Guide</i> for a description of each subset  _____</p>
LINE <i>range</i>	Specifies a single line number or range of lines. Line numbers display in columns 1 through 6 of Source View.
PARAGRAPH <i>name</i>	Specifies a PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. The entire paragraph or section is included.
asterisk (*)	Reuses the target of the previous HIGH command.
DATA <i>name</i>	Specifies a COBOL dataname or qualified COBOL dataname. Dataname refers to any valid COBOL reference for a data element. These subordinate operands apply to the dataname and can be used to further qualify the EXCLUDE command: REF, USE, MOD, DEF, ALIAS, NOALIAS, DIRECT, and INDIRECT. The defaults are REF, ALIAS, and DIRECT.
Ref	Excludes occurrences of the dataname where its value is defined, tested, used, set, or modified. This is the default value for the DATA name operand.
SQL <i>name</i>	Includes datanames that are DB2/SQL variables only.
IDMS <i>name</i>	Includes datanames that are IDMS variables only.
: <i>name</i>	Includes datanames that are COBOL variables only.
Ref	Includes occurrences of the dataname where its value is defined, tested, used, set, or modified. This is the default value for the DATA name operand.
Use	Excludes occurrences of the dataname where its value is being tested or used.

Operand	Description
Mod	Excludes occurrences of the dataname where its value is being set or modified.
Def	Includes definitions of the dataname in the DATA DIVISION
ALias	<p>Searches backward from the cursor position to the previous occurrence of the requested target.</p> <p>Includes occurrences of the aliases for the dataname. These are the valid aliases:</p> <ul style="list-style-type: none"> <li>• Parent - higher level group item</li> <li>• Child - lower level item</li> <li>• Rename/Redefinition - renamed, redefined, or 88 level items</li> </ul> <p>This is the default value for the DATA name operand.</p>
NOAlias	Ignores aliases for the specified dataname.
DIRect	Includes occurrences of the dataname (and aliases if specified) where it is directly tested, used, set, modified, or defined (based on the REF, USE, MOD, or DEF selection). The default for the DATA name operand is direct; however, if you specify the SIZE, VALUE, or LEVELS operand, INDIRECT is assumed.
INDirect	Includes any dataname indirectly affected by the specified dataname (and aliases if specified). You can further qualify the indirect data item by using the SIZE, VALUE, and LEVELS subordinate operands. These subordinate operands indicate the type of indirect reference to be located. SIZE and all levels are the defaults for the INDIRECT operand.
SIZE	Includes datanames that could be directly or indirectly affected if the size of the specified dataname were to be changed. SIZE is only valid with the INDIRECT operand. This is the default for the INDIRECT operand.
VALue	Includes occurrences of datanames that could be directly or indirectly affected by a change if the value of the specified dataname were to be changed. The LEVELS subordinate operand is not used with VALUE. VALUE is only valid with the INDIRECT operand.
LEVEls <i>n</i>	Includes all data items affected within the specified number of indirect levels. The VALUE subordinate operand is not used with LEVELS. All levels is the default for the LEVELS operand. LEVELS is only valid with the INDIRECT operand.

Operand	Description
PATTERN	Indicates the characters that follow are part of a string.
<i>string</i>	Specifies a string of alphanumeric or DBCS characters. If the pattern string contains blanks, it must be enclosed in single or double quotes. The pattern string can be further qualified by using the WORD, PREFIX, or SUFFIX subordinate operands. These subordinate operands describe how the pattern string is to be used.
X' <i>string</i> '	Specifies the hexadecimal string option. You can specify specific unprintable characters by giving the EBCDIC hexadecimal value. The string must be enclosed in single or double quotes.
T' <i>string</i> '	Specifies the text string option. A character string may be entered regardless of upper or lowercase by using the text option. The string must be enclosed in single or double quotes.
P' <i>string</i> '	Specifies the picture string option. A string profile may be entered instead of exact characters. The string must be enclosed in single or double quotes. Insight defines nine special characters for use in picture strings. These special characters can be combined with other characters.  These are the special characters: P'=' Any character P'-' Any nonblank character P'.' Any nondisplay character P'#' Any numeric character P'-' Any character P'@' Any character P'<' Any character P'>' Any character P'\$' Any character
WORD	Specifies the specified pattern string preceded and followed by any non-alphanumeric character (except a hyphen).
PREFIX	Specifies a word that begins with the specified pattern string.
SUFFIX	Specifies a word that ends with the specified pattern string.
Column1	Specifies the column number where the search is to begin.
Column2	Specifies the column number where the search is to end.

Operand	Description
Next	Searches forward from the current cursor position to the next occurrence of the requested target.
Prev	Searches backward from the current cursor position to the previous occurrence of the requested target.
Ffirst	Searches from the top of the source file to the first occurrence of the requested target.
LAst	Searches backward from the bottom of the source file to the first occurrence of the requested target.
All	Searches for all occurrences of the requested target and displays the number found in the short/long message field. This is the default value for the HIGH command.
IN	Restricts the HIGH command to the specified target type.

### Usage Notes

Source code lines can also be highlighted by selecting a Search pop-up on the Search pull-down. See the online help for more information.

To concatenate targets, place a plus sign (+) between the target names. For example, IO + CALL. These rules apply to concatenation:

- A concatenated dataname can be used wherever a dataname is valid. Dataname subordinate operands (REF, ALIAS, etc.) pertain to all datanames in a concatenated series.
- A concatenated set name can be used wherever a set name is valid. These sets can be concatenated:
  - LABEL
  - MARK
  - PERFRANGE
  - SUBSET
  - LINE
  - PROGRAM
  - PARAGRAPH
  - PATTERN
- The asterisk (\*) operand can be concatenated once with any number of other operands (e.g., \* + IE + MYSET); however, the \* operand cannot be concatenated to itself (\* + \* is invalid). The \* operand may appear in any order in the concatenated list.

For COBOL II Release 3 or later programs, you can qualify a data item, label, or program name that may be ambiguous or used multiple times by using OF followed by the program name. For example, if the label P120-READ exists in another program and also in the program VIAIDEM1, then it can be qualified as P120-READ OF VIAIDEM1.

Lines can be added to a highlighted set. For example, to see all PERFORM statements, then to add the conditional statements related to one of the PERFORM statements, use FINDXTND PERFORM, then use HIGH CONDITIONAL to add the additional lines.

To step through all IO statements, type HIGH IO NEXT. Keep repeating the command until the end of the program is located.

The H or HH line commands can also be used to highlight source code lines during a Source View session.

### *Example*

All lines containing IO statements are highlighted. Existing highlighted lines are not reset when this command is issued.

```
HIGH IOPF15
```

Next, type this command:

```
FX LAB IN HI
```

Only labels within the highlighted lines are highlighted. All other highlighted lines are reset.

## JUMP Command

JUMP 

### *Function*

The JUMP command is used to return to the Source View screen at the location corresponding to the cursor position on the Tree View screen.

### *Operands*

None.

### *Usage Notes*

The JUMP command is used in conjunction with the cursor position. Type JUMP in the command input area, position the cursor to the line containing the target of the jump, and press Enter.

The J line command can also be used to return to the Source View screen from the Tree View screen.

## KEYS Command



### Function

The KEYS command, entered with no operands, displays the Options - PF Key Definition pop-up. This screen is used to display and/or modify the current Insight PF key assignments.

The KEYS command may also save, restore, or modify the PF key assignments without displaying the Options - PF Key Definition pop-up.

### Operands

Operand	Description
SAVE	Saves the current PF Key definitions in a sequential or partitioned dataset. If you do not specify a dataset name, this is the default name: <i>TSOuserid.PPPnnnnn.VIAPFKEY</i> where <i>nnnnn</i> is a unique, system-generated number.
LOAD	Restores the PF Key definitions from the specified sequential or partitioned dataset. A dataset name must be specified for the LOAD operand.
<i>dsn</i>	Specifies a sequential dataset name. If you include quotes, the dataset name is used as is. If you do not include quotes, your TSO prefix or userID is appended to the front of the dataset name and VIAPFKEY is appended to the end of the dataset name. For example, if you type this command:  SAVE TEST1  the generated dataset name would be <i>TSOuserid.TEST1.VIAPFKEY</i> .

Operand	Description
<i>dsn(member)</i>	Specifies a partitioned dataset and member name. If you include quotes, the dataset name and member name are used as is. If you do not use quotes, your TSO prefix or userID is appended to the front of the dataset name. If the PDS member already exists, it is overwritten.
<i>PFnn</i>	<p>Provides a command line update to the PF key definition. For example, this command assigns the text RESET EXCLUDED to PF key 21:</p> <pre>KEYS PF21 RESET EXCLUDED</pre> <p>If the text to be assigned begins with an equals sign (=), you must enter two equal symbols separated by a space. If a null string is entered for the <i>PFnn</i> operand, the PF Key assignment reverts to the default value for the current ESW product.</p> <p>The script facility verifies that the PF Key number is valid.</p>

### Usage Notes

You can enter the KEYS command without operands on any Insight screen to display and/or modify the current PF key assignments. Values assigned to the Insight PF keys have no effect on other ISPF applications.

Use the SAVE and LOAD operands to save or restore PF Key settings. This feature allows you to use the KEYS command in scripts to set PF Key assignments as desired. This also provides a mechanism for multiple users to share a set of PF Key definitions.

The *PFnn* operand allows you to use the KEYS command in a script file to modify PF Key settings during script file execution.

You can also display the Options - PF Key Definition pop-up by selecting PF Keys on the Options pull-down. See the online help for more information about PF key assignments, the Options - PF Key Definition pop-up, and the Options pull-down.

## LEVELS Command

LEVELS   →

### Function

The LEVELS command redisplay the current Tree View or Structure View screen to the specified level of program hierarchy depth.

### Operands

Operand	Description
<i>number</i>	Redisplay the current screen to the level depth specified.
MAX	Redisplay the current screen to the maximum number of levels in the program.

### Usage Notes

You can also change levels from the View - Levels Request pop-up. To display the View - Levels Request pop-up, select View ▶ Level and press Enter.

The LEVELS command redisplay the current screen to the level specified. If there are less levels than the number specified, they expand to the requested level. If there are levels greater than the number specified, they are removed. The LEVELS command is only valid on the Tree View and Structure View screens.

## LIST Command



### Function

The LIST command is used to display the specified pop-up. List pop-ups are available for each operand shown above.

### Operands

Operands	Description
Blank	Displays the List Menu pop-up that allows you to select one of the other List pop-ups.
Calls	Displays the List - CALL Statements pop-up that shows all of the COBOL statements containing CALLs to other programs in the active program. This pop-up also classifies the CALLs into internal, static, and dynamic.
Equates	Displays the List - Equates pop-up that shows all of the equates for the active program.
Marks	Displays the List - User Marks pop-up that shows the marks for the active program.
Performs	Displays the List - Perform Range Names pop-up that shows all of the COBOL PERFORM ranges in the active program.
PRograms   PGms	<p>Displays the List - Program/Subprogram Names pop-up, that shows the nested program hierarchy structure.</p> <p>For COBOL II Release 3 or later programs, you can qualify a data item, label, or program name that may be ambiguous or used multiple times by using OF followed by the program name. For example, if the label P120-READ exists in another program and also in the program VIAIDEM1, then it can be qualified as P120-READ OF VIAIDEM1.</p>
Subsets	Displays the List - COBOL Subset Names pop-up that shows all subsets along with a brief description of each. Valid in Source View.

## Usage Notes

You can also access these pop-ups by selecting an action on the List pull-down.

## List Menu Pop-up

The List Menu pop-up is used to select the specified pop-up that displays pertinent information about the program being viewed. A screen similar to [Figure 101](#) displays when you type LIST without operands.

Figure 101 • List Menu Pop-up

```

                                List Menu
Select the list to be displayed. Then press Enter.
$  Keyword      Description
-----
-  CALLS       List CALLED programs and their mode
-  EQUATES     List EQUATES for the current program
-  MARKS       List MARKS for the current program
-  PERFORMS    List PERFORM ranges
-  PROGRAMS    List PROGRAMS and subPROGRAMS
-  SUBSET$     List COBOL SUBSET$ and their descriptions

```

Select any item on this pop-up by typing S to the left of the desired keyword. The pop-up for the selected option displays.

## LOCATE Command



### Function

The LOCATE command scrolls to the selectable label or entry that most closely matches the specified string. LOCATE is valid on the Source View screen, the File - AKR Directory pop-up, the File - AKR Program Selection pop-up, all List pop-ups except the List -Program/Subprogram Names pop-up, and the Trace Decision Options pop-up.

### Operands

Operands	Description
<i>string</i>	Specifies an alphanumeric or DBCS character string that locates a particular item on a list or directory type screen.
<i>.label</i>	Locates a line label entered in the line command area (columns 1 through 6) on the Source View screen.
<i>&amp;CURRENT</i>	Positions the Source View screen to the last location marked by the CURRENT command. If no CURRENT command has preceded the LOCATE &CURRENT command, the message &CURRENT NOT AVAIL displays and the screen is not repositioned. The &CURRENT operand is only valid on the Source View screen.

### Usage Notes

You can specify a character string when an exact program name or list item is not known. The LOCATE command displays the item that most closely matches the specified character string. Matching is done alphabetically.

## Examples

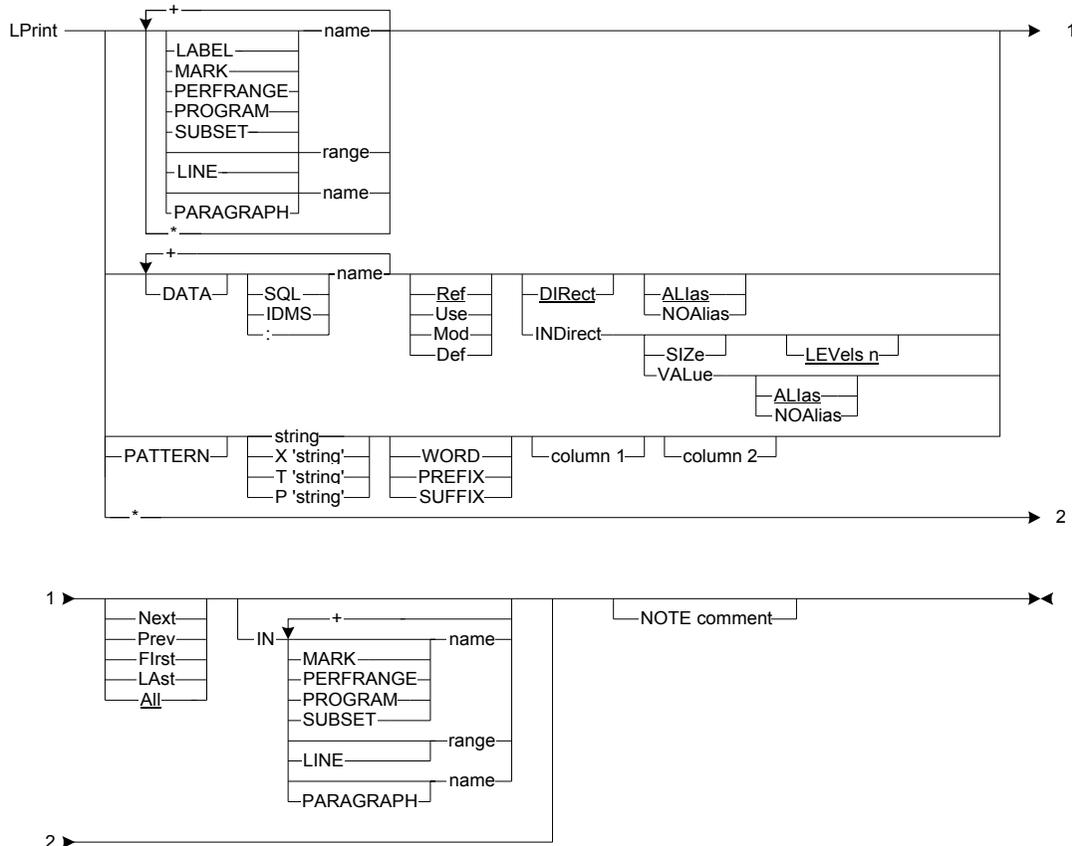
**Example 1.** This command displays the line containing an item that matches or follows S:

```
LOCATE S
```

**Example 2.** This command positions the display to the location marked by the most recent CURRENT command.

```
LOCATE &CURRENT
```

## LPRINT Command



### Function

The LPRINT command is used to copy lines containing the specified target to the List file. The List file is processed on the Options - Log/List/Punch Definition pop-up.

### Operands

Operand	Description
LABEL <i>name</i>	Specifies the paragraph or section name of the PROCEDURE DIVISION or the literals PROCEDURE or the PROC where the FLOW command is to begin.

Operand	Description																		
MARK <i>name</i>	<p>Specifies a 1 to 10 alphanumeric character name given to a set or path using the COPY, MARK, MERGE, or RENAME commands, or one of these system-generated paths:</p> <p style="padding-left: 40px;">System generated path: Created by:</p> <p style="padding-left: 40px;">TRACK   TRK      A TRACE command.</p> <p style="padding-left: 40px;">NETWORK   NET    A FLOW command.</p> <p style="padding-left: 40px;">SUBNET<sub><i>n</i></sub>   SUB<sub><i>n</i></sub>    A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (<i>n</i><sup>th</sup> result).</p> <p><b>Note:</b> _____ The TRACK, NETWORK, and SUBNET paths can also be created using the Logic Order Search pop-ups (see "<a href="#">Logic Pull-down</a>" on <a href="#">page 101</a>). _____</p>																		
PERFRANGE <i>name</i>	<p>Specifies the name in a PERFORM statement including all statements executed as a result of a PERFORM statement, or the name of any section contained in the Declaratives.</p> <p>This operand is valid in Source View only</p>																		
PROGRAM <i>name</i>	<p>Specifies the name of the main program or any nested program representing all the code contained in the program. This includes all the programs physically nested inside the specified program.</p>																		
SUBSET <i>name</i>	<p>Specifies a predefined subset. These are the valid subsets:</p> <table style="margin-left: 40px;"> <tbody> <tr> <td>ASsignment</td> <td>DEBug</td> <td>IO</td> </tr> <tr> <td>CALL</td> <td>DEFinition</td> <td>LABel</td> </tr> <tr> <td>CIcs</td> <td>DIRective</td> <td>MAINline</td> </tr> <tr> <td>COBOLII</td> <td>DIVision</td> <td>MATH</td> </tr> <tr> <td>COBOL/370</td> <td>DL/I   DL/1</td> <td>Output</td> </tr> <tr> <td>COMment</td> <td>DML</td> <td>PARagraph</td> </tr> </tbody> </table>	ASsignment	DEBug	IO	CALL	DEFinition	LABel	CIcs	DIRective	MAINline	COBOLII	DIVision	MATH	COBOL/370	DL/I   DL/1	Output	COMment	DML	PARagraph
ASsignment	DEBug	IO																	
CALL	DEFinition	LABel																	
CIcs	DIRective	MAINline																	
COBOLII	DIVision	MATH																	
COBOL/370	DL/I   DL/1	Output																	
COMment	DML	PARagraph																	

Operand	Description
	<p>CONditional    ENtry            PERform</p> <p>COPy            EXIt   PGM       ExitRETurn</p> <p>DB2/SQL        FALLthrough     SEction</p> <p>DDL            GOto             SORTMerge</p> <p>DEAD           IDMS             STructure</p> <p>DEADCode      INclude          TESTed</p> <p>DEADData      Input             UNTested</p> <p>A screen subset:</p> <p>Highlighted   HI</p> <p>NONHighlighted   NHI</p> <p>Excluded   X</p> <p>NONExcluded   NX</p> <p>Tagged lines subsets of tags in columns 73 through 80.</p> <p><b>Note:</b> _____</p> <p>See the <i>ASG-Insight User's Guide</i> for a description of each subset</p> <p>_____</p>
LINE <i>range</i>	Specifies a single line number or range of lines. Line numbers display in columns 1 through 6 of Source View.
PARAGRAPH <i>name</i>	Specifies a PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. The entire paragraph or section is included.
DATA <i>name</i>	Specifies a COBOL dataname or qualified COBOL dataname. Dataname refers to any valid COBOL reference for a data element. These subordinate operands apply to the dataname and can be used to further qualify the EXCLUDE command: REF, USE, MOD, DEF, ALIAS, NOALIAS, DIRECT, and INDIRECT. The defaults are REF, ALIAS, and DIRECT.
SQL <i>name</i>	Includes datanames that are DB2/SQL variables only.
IDMS <i>name</i>	Includes datanames that are IDMS variables only.
: <i>name</i>	Includes datanames that are COBOL variables only.
Ref	Includes occurrences of the dataname where its value is defined, tested, used, set, or modified. This is the default value for the DATA name operand.

Operand	Description
Use	Excludes occurrences of the dataname where its value is being tested or used.
Mod	Excludes occurrences of the dataname where its value is being set or modified.
Def	Includes definitions of the dataname in the DATA DIVISION
ALias	<p>Searches backward from the cursor position to the previous occurrence of the requested target.</p> <p>Includes occurrences of the aliases for the dataname. These are the valid aliases:</p> <ul style="list-style-type: none"> <li>• Parent - higher level group item</li> <li>• Child - lower level item</li> <li>• Rename/Redefinition - renamed, redefined, or 88 level items</li> </ul> <p>This is the default value for the DATA name operand.</p>
NOAlias	Ignores aliases for the specified dataname.
DIRect	Includes occurrences of the dataname (and aliases if specified) where it is directly tested, used, set, modified, or defined (based on the REF, USE, MOD, or DEF selection). The default for the DATA name operand is direct; however, if you specify the SIZE, VALUE, or LEVELS operand, INDIRECT is assumed.
INDirect	Includes any dataname indirectly affected by the specified dataname (and aliases if specified). You can further qualify the indirect data item using the SIZE, VALUE, and LEVELS subordinate operands. These subordinate operands indicate the type of indirect reference to be located. SIZE and all levels are the defaults for the INDIRECT operand.
SIZE	Includes datanames that could be directly or indirectly affected if the size of the specified dataname were to be changed. SIZE is only valid with the INDIRECT operand. This is the default for the INDIRECT operand.
VALue	Includes occurrences of datanames that could be directly or indirectly affected by a change if the value of the specified dataname were to be changed. The LEVELS subordinate operand is not used with VALUE. VALUE is only valid with the INDIRECT operand.

Operand	Description
LEVels <i>n</i>	Includes all data items affected within the specified number of indirect levels. The VALUE subordinate operand is not used with LEVELS. All levels is the default for the LEVELS operand. LEVELS is only valid with the INDIRECT operand.
PATTERN	Indicates the characters that follow are part of a string.
<i>string</i>	Specifies a string of alphanumeric or DBCS characters. If the pattern string contains blanks, it must be enclosed in single or double quotes. The pattern string can be further qualified using the WORD, PREFIX, or SUFFIX subordinate operands. These subordinate operands describe how the pattern string is to be used.
X' <i>string</i> '	Specifies the hexadecimal string option. Specific unprintable characters may be specified by giving the EBCDIC hexadecimal value. The string must be enclosed in single or double quotes.
T' <i>string</i> '	Specifies the text string option. A character string may be entered regardless of upper or lowercase by using the text option. The string must be enclosed in single or double quotes.
P' <i>string</i> '	Specifies the picture string option. You can enter a string profile instead of exact characters. The string must be enclosed in single or double quotes. Insight defines nine special characters for use in picture strings. These special characters can be combined with other characters.  These are the special characters: P'=' Any character P'-' Any nonblank character P'.' Any nondisplay character P'#' Any numeric character P'-' Any character P'@' Any character P'<' Any character P'>' Any character P'\$' Any character
WORD	Specifies the specified pattern string preceded and followed by any non-alphanumeric character (except a hyphen).
PREFIX	Specifies a word that begins with the specified pattern string.

Operand	Description
SUFFIX	Specifies a word that ends with the specified pattern string.
Column1	Specifies the column number where the search is to begin.
Column2	Specifies the column number where the search is to end.
asterisk (*)	Causes the entire virtual screen (all data that can be viewed by scrolling forward and backward) to be copied to the List file. All excluded lines are copied to the List file as excluded lines (as they appear on the screen at the time the LPRINT * command is entered).
Next	Searches forward from the current cursor position to the next occurrence of the requested target.
Prev	Searches backward from the current cursor position to the previous occurrence of the requested target.
Ffirst	Searches from the top of the source file to the first occurrence of the requested target.
LAst	Searches backward from the bottom of the source file to the first occurrence of the requested target.
All	Searches for all occurrences of the requested target and displays the number found in the short/long message field. This is the default value for the HIGH command.
IN	Restricts the HIGH command to the specified target type.
NOTE comment	Places descriptive comment lines in the List file. These comments are included in the printed output from the List file. Text of the comment may be a maximum of 50 alphanumeric or 23 DBCS characters.

### Usage Notes

Lines can also be copied to the List file by selecting a Search pop-up on the Search pull-down. See the online help for more information.

To concatenate targets, place a plus sign (+) between the target names. For example, IO + CALL. These rules apply to concatenation:

- A concatenated dataname can be used wherever a dataname is valid. Dataname subordinate operands (REF, ALIAS, etc.) pertain to all datanames in a concatenated series.

- A concatenated set name can be used wherever a set name is valid. These sets can be concatenated:
  - LABEL
  - MARK
  - PERFRANGE
  - SUBSET
  - LINE
  - PROGRAM
  - PARAGRAPH

For COBOL II Release 3 or later programs, you can qualify a data item, label, or program name that may be ambiguous or used multiple times by using OF followed by the program name. For example, if the label P120-READ exists in another program and also in the program VIAIDEM1, then it can be qualified as P120-READ OF VIAIDEM1.

LPRINT copies the specified target to the List file for subsequent printing.

LPRINT \* can be entered on these pop-ups and the Tree View and Source View screen to copy them to the List file:

- List - CALL Statements
- List - Equates
- List - User Marks
- List - Perform Range Names
- List - Program/Subprogram Names
- List - COBOL Subset Names
- View - Paragraph Cross Reference

## *Examples*

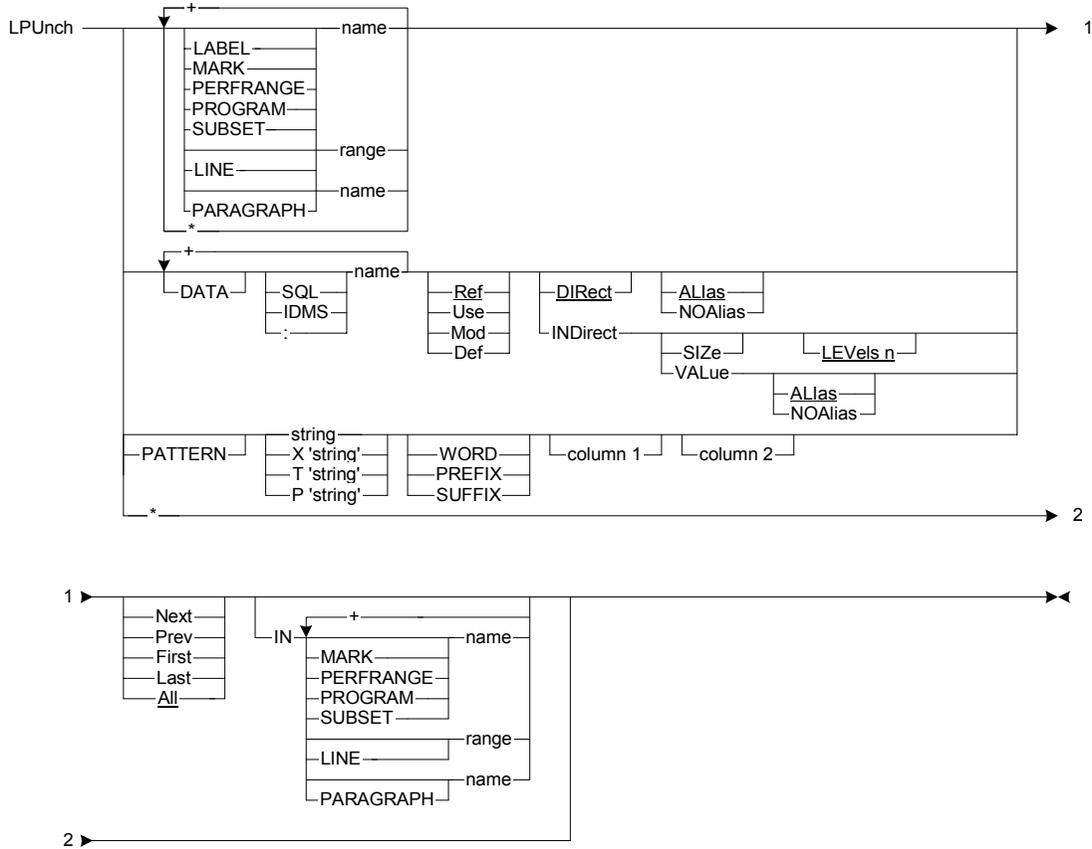
**Example 1.** To copy all lines containing the characters ZIP for subsequent printing, type this command:

```
LPRINT ZIP
```

**Example 2.** To copy the entire source file to the List file, type this command:

```
LPRINT *
```

## LPUNCH Command



### Function

The LPUNCH command is used to copy lines containing the specified target to the Punch file for subsequent processing. The Punch file is processed on the Options - Log/List/Punch Definition pop-up.

### Operands

Operand	Description
LABEL <i>name</i>	Specifies the paragraph or section name of the PROCEDURE DIVISION or the literals PROCEDURE or PROC where the FLOW command is to begin.
MARK <i>name</i>	Specifies a 1 to 10 alphanumeric character name given to a set or path using the COPY, MARK, MERGE, or RENAME commands, or one of these system-generated paths:

Operand	Description																																							
	<p>System generated path: Created by:</p> <p>TRACK   TRK A TRACE command.</p> <p>NETWORK   NET A FLOW command.</p> <p>SUBNET<sub>n</sub>   SUB<sub>n</sub> A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (n<sup>th</sup> result).</p>																																							
	<p><b>Note:</b></p> <p>The TRACK, NETWORK, and SUBNET paths can also be created using the Logic Order Search pop-ups (see <a href="#">"Logic Pull-down" on page 101</a>).</p>																																							
PERFRANGE <i>name</i>	<p>Specifies the name in a PERFORM statement including all statements executed as a result of a PERFORM statement, or the name of any section contained in the Declaratives.</p> <p>This operand is valid in Source View only</p>																																							
PROGRAM <i>name</i>	<p>Specifies the name of the main program or any nested program representing all the code contained in the program. This includes all the programs physically nested inside the specified program.</p>																																							
SUBSET <i>name</i>	<p>Specifies a predefined subset. These are the valid subsets:</p> <table border="0"> <tr> <td>ASsignment</td> <td>DEBUg</td> <td>IO</td> </tr> <tr> <td>CALL</td> <td>DEFinition</td> <td>LABEL</td> </tr> <tr> <td>CIcs</td> <td>DIRective</td> <td>MAINline</td> </tr> <tr> <td>COBOLII</td> <td>DIVision</td> <td>MATH</td> </tr> <tr> <td>COBOL/370</td> <td>DL/I   DL/1</td> <td>Output</td> </tr> <tr> <td>COMment</td> <td>DML</td> <td>PARagraph</td> </tr> <tr> <td>CONditional</td> <td>ENtry</td> <td>PERform</td> </tr> <tr> <td>COPY</td> <td>EXIt   PGM</td> <td>ExitRETurn</td> </tr> <tr> <td>DB2/SQL</td> <td>FALLthrough</td> <td>SECTion</td> </tr> <tr> <td>DDL</td> <td>GOto</td> <td>SORTMerge</td> </tr> <tr> <td>DEAD</td> <td>IDMS</td> <td>STRucture</td> </tr> <tr> <td>DEADCode</td> <td>INClude</td> <td>TESTed</td> </tr> <tr> <td>DEADData</td> <td>Input</td> <td>UNTested</td> </tr> </table>	ASsignment	DEBUg	IO	CALL	DEFinition	LABEL	CIcs	DIRective	MAINline	COBOLII	DIVision	MATH	COBOL/370	DL/I   DL/1	Output	COMment	DML	PARagraph	CONditional	ENtry	PERform	COPY	EXIt   PGM	ExitRETurn	DB2/SQL	FALLthrough	SECTion	DDL	GOto	SORTMerge	DEAD	IDMS	STRucture	DEADCode	INClude	TESTed	DEADData	Input	UNTested
ASsignment	DEBUg	IO																																						
CALL	DEFinition	LABEL																																						
CIcs	DIRective	MAINline																																						
COBOLII	DIVision	MATH																																						
COBOL/370	DL/I   DL/1	Output																																						
COMment	DML	PARagraph																																						
CONditional	ENtry	PERform																																						
COPY	EXIt   PGM	ExitRETurn																																						
DB2/SQL	FALLthrough	SECTion																																						
DDL	GOto	SORTMerge																																						
DEAD	IDMS	STRucture																																						
DEADCode	INClude	TESTed																																						
DEADData	Input	UNTested																																						

Operand	Description
	<p>A screen subset:</p> <p>Highlighted   HI  NONHighlighted   NHI  Excluded   X  NONExcluded   NX</p> <p>Tagged lines subsets of tags in columns 73 through 80.</p> <p><b>Note:</b> _____  See the <i>ASG-Insight User's Guide</i> for a description of each subset  _____</p>
LINE <i>range</i>	Specifies a single line number or range of lines. Line numbers display in columns 1 through 6 of Source View.
PARAGRAPH <i>name</i>	Specifies a PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. The entire paragraph or section is included.
DATA <i>name</i>	Specifies a COBOL dataname or qualified COBOL dataname. Dataname refers to any valid COBOL reference for a data element. These subordinate operands apply to the dataname and can be used to further qualify the EXCLUDE command: REF, USE, MOD, DEF, ALIAS, NOALIAS, DIRECT, and INDIRECT. The defaults are REF, ALIAS, and DIRECT.
SQL <i>name</i>	Includes datanames that are DB2/SQL variables only.
IDMS <i>name</i>	Includes datanames that are IDMS variables only.
: <i>name</i>	Includes datanames that are COBOL variables only.
Ref	Includes occurrences of the dataname where its value is defined, tested, used, set, or modified. This is the default value for the DATA name operand.
Use	Excludes occurrences of the dataname where its value is being tested or used.
Mod	Excludes occurrences of the dataname where its value is being set or modified.
Def	Includes definitions of the dataname in the DATA DIVISION

Operand	Description
ALias	<p>Searches backward from the cursor position to the previous occurrence of the requested target.</p> <p>Includes occurrences of the aliases for the dataname. These are the valid aliases:</p> <ul style="list-style-type: none"> <li>• Parent - higher level group item</li> <li>• Child - lower level item</li> <li>• Rename/Redefinition - renamed, redefined, or 88 level items</li> </ul> <p>This is the default value for the DATA name operand.</p>
NOAlias	<p>Ignores aliases for the specified dataname.</p>
DIRect	<p>Includes occurrences of the dataname (and aliases if specified) where it is directly tested, used, set, modified, or defined (based on the REF, USE, MOD, or DEF selection). The default for the DATA name operand is direct; however, if you specify the SIZE, VALUE, or LEVELS operand, INDIRECT is assumed.</p>
INDirect	<p>Includes any dataname indirectly affected by the specified dataname (and aliases if specified). You can further qualify the indirect data item using the SIZE, VALUE, and LEVELS subordinate operands. These subordinate operands indicate the type of indirect reference to be located. SIZE and all levels are the defaults for the INDIRECT operand.</p>
SIZE	<p>Includes datanames that could be directly or indirectly affected if the size of the specified dataname were to be changed. SIZE is only valid with the INDIRECT operand. This is the default for the INDIRECT operand.</p>
VALue	<p>Includes occurrences of datanames that could be directly or indirectly affected by a change if the value of the specified dataname were to be changed. The LEVELS subordinate operand is not used with VALUE. VALUE is only valid with the INDIRECT operand.</p>
LEVels <i>n</i>	<p>Includes all data items affected within the specified number of indirect levels. The VALUE subordinate operand is not used with LEVELS. All levels is the default for the LEVELS operand. LEVELS is only valid with the INDIRECT operand.</p>
PATTERN	<p>Indicates the characters that follow are part of a string.</p>

Operand	Description
<i>string</i>	Specifies a string of alphanumeric or DBCS characters. If the pattern string contains blanks, it must be enclosed in single or double quotes. You can further qualify the pattern string using the WORD, PREFIX, or SUFFIX subordinate operands. These subordinate operands describe how the pattern string is to be used:
X' <i>string</i> '	Specifies the hexadecimal string option. You can specify specific unprintable characters by giving the EBCDIC hexadecimal value. The string must be enclosed in single or double quotes.
T' <i>string</i> '	Specifies the text string option. Use the text option to enter a character string regardless of upper or lowercase. The string must be enclosed in single or double quotes.
P' <i>string</i> '	Specifies the picture string option. You can enter a string profile instead of exact characters. The string must be enclosed in single or double quotes. Insight defines nine special characters for use in picture strings, and you can combine these special characters with other characters.  These are the special characters: P'=' Any character P'-' Any nonblank character P'.' Any nondisplay character P'#' Any numeric character  P'-' Any character P'@' Any character P'<' Any character P'>' Any character P'\$' Any character
WORD	Specifies the specified pattern string preceded and followed by any non-alphanumeric character (except a hyphen).
PREFIX	Specifies a word that begins with the specified pattern string.
SUFFIX	Specifies a word that ends with the specified pattern string.
Column1	Specifies the column number where the search is to begin.
Column2	Specifies the column number where the search is to end.

Operand	Description
asterisk (*)	Causes the entire virtual screen (all data that can be viewed by scrolling forward and backward) to be copied to the List file. All excluded lines are copied to the List file as excluded lines (as they appear on the screen at the time the LPRINT * command is entered).
Next	Searches forward from the current cursor position to the next occurrence of the requested target.
Prev	Searches backward from the current cursor position to the previous occurrence of the requested target.
First	Searches from the top of the source file to the first occurrence of the requested target.
Last	Searches backward from the bottom of the source file to the first occurrence of the requested target.
All	Searches for all occurrences of the requested target and displays the number found in the short/long message field. This is the default value for the HIGH command.
IN	Restricts the HIGH command to the specified target type.

### Usage Notes

You can also copy lines to the Punch file by selecting a Search pop-up on the Search pull-down. See the online help for more information.

LPUNCH copies the specified targets to the Punch file for subsequent processing.

To concatenate targets, place a plus sign (+) between the target names. For example, IO + CALL. These rules apply to concatenation:

- A concatenated dataname can be used wherever a dataname is valid. Dataname subordinate operands (REF, ALIAS, etc.) pertain to all datanames in a concatenated series.
- A concatenated set name can be used wherever a set name is valid. These sets can be concatenated:
  - LABEL
  - MARK
  - PERFRANGE
  - SUBSET
  - LINE
  - PROGRAM
  - PARAGRAPH

For COBOL II Release 3 or later programs, a data item, label, or program name that may be ambiguous or used multiple times can be qualified by using OF followed by the program name. For example, if the label P120-READ exists in another program and also in the program VIAIDEM1, then it can be qualified as P120-READ OF VIAIDEM1.

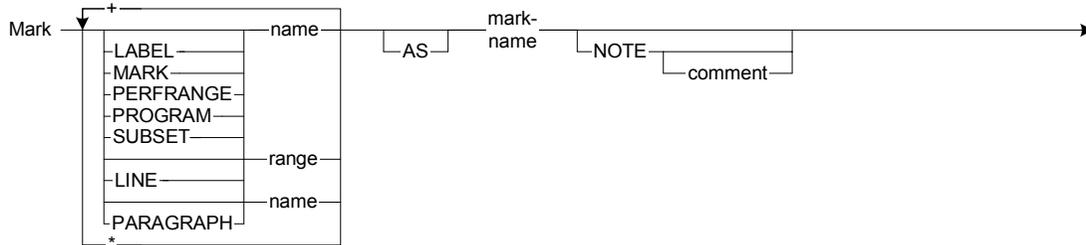
LPUNCH \* can be entered on Tree View and Source View screens.

**Example**

This command copies all the input and output statements of the current program into the Punch file.

```
LPUNCH IO
```

## MARK Command



### Function

The MARK command is used to save the requested target as a mark set or path. An optional description can be included if desired.

### Operands

Operand	Description								
LABEL <i>name</i>	Specifies the paragraph or section name of the PROCEDURE DIVISION or the literals PROCEDURE or PROC where the FLOW command is to begin.								
MARK <i>name</i>	Specifies a 1 to 10 alphanumeric character name given to a set or path using the COPY, MARK, MERGE, or RENAME commands, or one of these system-generated paths: <table border="0"> <tr> <td>System generated path:</td> <td>Created by:</td> </tr> <tr> <td>TRACK   TRK</td> <td>A TRACE command.</td> </tr> <tr> <td>NETWORK   NET</td> <td>A FLOW command.</td> </tr> <tr> <td>SUBNET<sub>n</sub>   SUB<sub>n</sub></td> <td>A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (nth result).</td> </tr> </table>	System generated path:	Created by:	TRACK   TRK	A TRACE command.	NETWORK   NET	A FLOW command.	SUBNET <sub>n</sub>   SUB <sub>n</sub>	A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (nth result).
System generated path:	Created by:								
TRACK   TRK	A TRACE command.								
NETWORK   NET	A FLOW command.								
SUBNET <sub>n</sub>   SUB <sub>n</sub>	A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (nth result).								
	<b>Note:</b> The TRACK, NETWORK, and SUBNET paths can also be created using the Logic Order Search pop-ups (see <a href="#">"Logic Pull-down" on page 101</a> ).								
PERFRANGE <i>name</i>	Specifies the name specified in a PERFORM statement including all statements executed as a result of a PERFORM statement, or the name of any section contained in the Declaratives.  This operand is valid in Source View only								

Operand	Description																																							
PROGRAM <i>name</i>	Specifies the name of the main program or any nested program representing all the code contained in the program. This includes all the programs physically nested inside the specified program.																																							
SUBSET <i>name</i>	<p>Specifies a predefined subset. These are the valid subsets:</p> <table border="0"> <tr> <td>ASsignment</td> <td>DEBUg</td> <td>IO</td> </tr> <tr> <td>CALL</td> <td>DEFinition</td> <td>LABEL</td> </tr> <tr> <td>CIcs</td> <td>DIRective</td> <td>MAINline</td> </tr> <tr> <td>COBOLII</td> <td>DIVision</td> <td>MATH</td> </tr> <tr> <td>COBOL/370</td> <td>DL/I   DL/1</td> <td>Output</td> </tr> <tr> <td>COMment</td> <td>DML</td> <td>PARagraph</td> </tr> <tr> <td>CONditional</td> <td>ENtry</td> <td>PERform</td> </tr> <tr> <td>COPy</td> <td>EXIt   PGM</td> <td>ExitRETurn</td> </tr> <tr> <td>DB2/SQL</td> <td>FALLthrough</td> <td>SECTION</td> </tr> <tr> <td>DDL</td> <td>GOto</td> <td>SORTMerge</td> </tr> <tr> <td>DEAD</td> <td>IDMS</td> <td>STructure</td> </tr> <tr> <td>DEADCode</td> <td>INclude</td> <td>TESTed</td> </tr> <tr> <td>DEADData</td> <td>Input</td> <td>UNTested</td> </tr> </table> <p>A screen subset:</p> <p>Highlighted   HI  NONHighlighted   NHI  Excluded   X  NONExcluded   NX</p> <p>Tagged lines subsets of tags in columns 73 through 80.</p> <p><b>Note:</b> _____  See the <i>ASG-Insight User's Guide</i> for a description of each subset  _____</p>	ASsignment	DEBUg	IO	CALL	DEFinition	LABEL	CIcs	DIRective	MAINline	COBOLII	DIVision	MATH	COBOL/370	DL/I   DL/1	Output	COMment	DML	PARagraph	CONditional	ENtry	PERform	COPy	EXIt   PGM	ExitRETurn	DB2/SQL	FALLthrough	SECTION	DDL	GOto	SORTMerge	DEAD	IDMS	STructure	DEADCode	INclude	TESTed	DEADData	Input	UNTested
ASsignment	DEBUg	IO																																						
CALL	DEFinition	LABEL																																						
CIcs	DIRective	MAINline																																						
COBOLII	DIVision	MATH																																						
COBOL/370	DL/I   DL/1	Output																																						
COMment	DML	PARagraph																																						
CONditional	ENtry	PERform																																						
COPy	EXIt   PGM	ExitRETurn																																						
DB2/SQL	FALLthrough	SECTION																																						
DDL	GOto	SORTMerge																																						
DEAD	IDMS	STructure																																						
DEADCode	INclude	TESTed																																						
DEADData	Input	UNTested																																						
LINE <i>range</i>	Specifies a single line number or range of lines. Line numbers display in columns 1 through 6 of Source View.																																							
PARAGRAPH <i>name</i>	Specifies a PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. The entire paragraph or section is included.																																							

Operand	Description
AS	Specifies an optional keyword indicating the mark name that follows is the name to be assigned to the target.
<i>mark-name</i>	<p>A name assigned to a path or set of lines using the MARK command or the Options - Scratchpad Mark pop-up. The specified name must meet these requirements:</p> <ul style="list-style-type: none"> <li>• A maximum of 10 alphanumeric characters, including hyphens. Names longer than 10 characters truncate.</li> <li>• Must begin with an alphabetic character.</li> </ul>
NOTE comment	Provides an optional description about the new mark name. Comment text can be a maximum of 50 alphanumeric or 23 DBCS characters. If the NOTE comment operand is not entered, the existing comment is copied to the new mark name. If NOTE is entered without comment text, the existing comment is not copied to the new mark name.

### Usage Notes

You can also create, copy, delete, merge, and rename mark sets and paths by selecting one of the options on the Options - Scratchpad pop-up. See the online help for more information.

To concatenate targets, place a plus sign (+) between the target names. For example, IO+CALL. A concatenated set name can be used wherever a set name is valid. These sets can be concatenated:

- LABEL
- MARK
- PERFRANGE
- SUBSET
- LINE
- PROGRAM
- PARAGRAPH

For COBOL II Release 3 or later programs, you can qualify a data item, label, or program name that may be ambiguous or used multiple times by using OF followed by the program name. For example, if the label P120-READ exists in another program and also in the program VIAIDEM1, then it can be qualified as P120-READ OF VIAIDEM1.

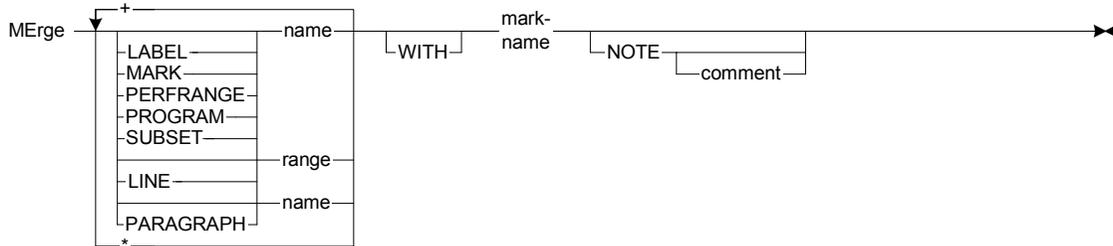
The target of a MARK command can be saved multiple times under different names.

### Example

This command captures the system-generated path called TRACK and is named FICA (both paths exist).

```
MARK TRACK AS FICA
```

## MERGE Command



### Function

The MERGE command adds the lines from the specified target to the specified mark name. If the specified mark name does not exist, it is created. A comment included with the MERGE command replaces an existing one.

### Operands

Operand	Description								
LABEL <i>name</i>	Specifies the paragraph or section name of the PROCEDURE DIVISION or the literals PROCEDURE or PROC where the FLOW command is to begin.								
MARK <i>name</i>	Specifies a 1 to 10 alphanumeric character name given to a set or path using the COPY, MARK, MERGE, or RENAME commands, or one of these system-generated paths: <table border="0" style="margin-left: 20px;"> <tr> <td>System generated path:</td> <td>Created by:</td> </tr> <tr> <td>TRACK   TRK</td> <td>A TRACE command.</td> </tr> <tr> <td>NETWORK   NET</td> <td>A FLOW command.</td> </tr> <tr> <td>SUBNET<sub><i>n</i></sub>   SUB<sub><i>n</i></sub></td> <td>A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (<i>n</i>th result).</td> </tr> </table>	System generated path:	Created by:	TRACK   TRK	A TRACE command.	NETWORK   NET	A FLOW command.	SUBNET <sub><i>n</i></sub>   SUB <sub><i>n</i></sub>	A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results ( <i>n</i> th result).
System generated path:	Created by:								
TRACK   TRK	A TRACE command.								
NETWORK   NET	A FLOW command.								
SUBNET <sub><i>n</i></sub>   SUB <sub><i>n</i></sub>	A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results ( <i>n</i> th result).								
<b>Note:</b>									
The TRACK, NETWORK, and SUBNET paths can also be created using the Logic Order Search pop-ups (see <a href="#">"Logic Pull-down" on page 101</a> ).									

Operand	Description																																							
PERFRANGE <i>name</i>	The name specified in a PERFORM statement including all statements executed as a result of a PERFORM statement, or the name of any section contained in the Declaratives. This operand is valid in Source View only																																							
PROGRAM <i>name</i>	The name of the main program or any nested program representing all the code contained in the program. This includes all the programs physically nested inside the specified program.																																							
SUBSET <i>name</i>	<p>Specifies a predefined subset. These are the valid subsets:</p> <table border="0"> <tr> <td>ASsignment</td> <td>DEBug</td> <td>IO</td> </tr> <tr> <td>CALL</td> <td>DEFinition</td> <td>LABEL</td> </tr> <tr> <td>CIcs</td> <td>DIRective</td> <td>MAINline</td> </tr> <tr> <td>COBOLII</td> <td>DIVision</td> <td>MATH</td> </tr> <tr> <td>COBOL/370</td> <td>DL/I   DL/1</td> <td>Output</td> </tr> <tr> <td>COMment</td> <td>DML</td> <td>PARagraph</td> </tr> <tr> <td>CONditional</td> <td>ENtry</td> <td>PERform</td> </tr> <tr> <td>COPY</td> <td>EXIt   PGM</td> <td>ExitRETurn</td> </tr> <tr> <td>DB2/SQL</td> <td>FALLthrough</td> <td>SECTION</td> </tr> <tr> <td>DDL</td> <td>GOto</td> <td>SORTMerge</td> </tr> <tr> <td>DEAD</td> <td>IDMS</td> <td>STRucture</td> </tr> <tr> <td>DEADCode</td> <td>INclude</td> <td>TESTed</td> </tr> <tr> <td>DEADData</td> <td>Input</td> <td>UNTested</td> </tr> </table> <p>A screen subset:            Highlighted   HI            NONHighlighted   NHI            Excluded   X            NONExcluded   NX</p> <p>Tagged lines subsets of tags in columns 73 through 80.</p> <p><b>Note:</b> _____            See the <i>ASG-Insight User's Guide</i> for a description of each subset            _____</p>	ASsignment	DEBug	IO	CALL	DEFinition	LABEL	CIcs	DIRective	MAINline	COBOLII	DIVision	MATH	COBOL/370	DL/I   DL/1	Output	COMment	DML	PARagraph	CONditional	ENtry	PERform	COPY	EXIt   PGM	ExitRETurn	DB2/SQL	FALLthrough	SECTION	DDL	GOto	SORTMerge	DEAD	IDMS	STRucture	DEADCode	INclude	TESTed	DEADData	Input	UNTested
ASsignment	DEBug	IO																																						
CALL	DEFinition	LABEL																																						
CIcs	DIRective	MAINline																																						
COBOLII	DIVision	MATH																																						
COBOL/370	DL/I   DL/1	Output																																						
COMment	DML	PARagraph																																						
CONditional	ENtry	PERform																																						
COPY	EXIt   PGM	ExitRETurn																																						
DB2/SQL	FALLthrough	SECTION																																						
DDL	GOto	SORTMerge																																						
DEAD	IDMS	STRucture																																						
DEADCode	INclude	TESTed																																						
DEADData	Input	UNTested																																						
LINE <i>range</i>	Specifies a single line number or range of lines. Line numbers display in columns 1 through 6 of Source View.																																							
PARAGRAPH <i>name</i>	Specifies a PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. The entire paragraph or section is included.																																							

Operand	Description
WITH	Specifies an optional keyword indicating the mark name that follows is to be merged with the specified target.
<i>mark-name</i>	Specifies a name assigned to a path or set of lines using the MARK command or the Options - Scratchpad Mark pop-up. The specified name must meet these requirements: <ul style="list-style-type: none"> <li>• A maximum of 10 alphanumeric characters, including hyphens. Names longer than 10 characters truncate.</li> <li>• Must begin with an alphabetic character.</li> </ul>
NOTE comment	Provides an optional description about the name. Comment text can be a maximum of 50 alphanumeric or 23 DBCS characters. If you do not enter the NOTE comment operand, the existing comment is copied to the new mark name. If you enter NOTE without comment text, the existing comment is not copied to the new mark name.

### Usage Notes

You can also merge paths and sets on the Options - Scratchpad Merge pop-up. See the online help for more information. A set of lines can be merged with another set of lines, or a path can be merged to a set of lines.

To concatenate targets, place a plus sign (+) between the target names. For example, IO+CALL. A concatenated set name can be used wherever a set name is valid. These sets can be concatenated:

- LABEL
- MARK
- PERFRANGE
- SUBSET
- LINE
- PROGRAM
- PARAGRAPH

For COBOL II Release 3 or later programs, you can qualify a data item, label, or program name that may be ambiguous or used multiple times by using OF followed by the program name. For example, if the label P120-READ exists in another program and also in the program VIAIDEM1, then it can be qualified as P120-READ OF VIAIDEM1.

### Example

The command below merges the contents of TRACK with the set of lines called FICA.

```
MERGE TRACK WITH FICA
```

## PARMDEF Command

PARMDEF | PDEF

---

### *Function*

The PARMDEF command displays the Options - Product Parameters pop-up. The Options - Product Parameters pop-up is used to set parameters that affect the online operation of Insight.

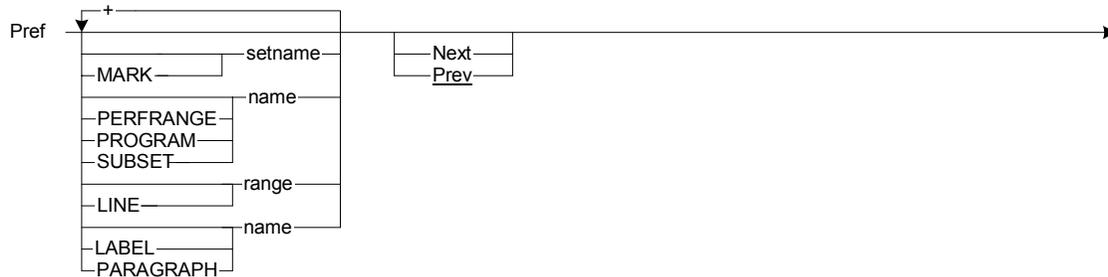
### *Operands*

None.

### *Usage Notes*

You can also display the Options - Product Parameters pop-up by selecting Options ▶ Product parameters.

## PREF Command



### Function

The PREF command is used to display the View - Paragraph Cross Reference pop-up for the requested target. The View - Paragraph Cross Reference pop-up shows how control is transferred to or from the target paragraphs.

### Operands

Operand	Description
Blank	If no operand is entered, the target paragraph set is determined by the cursor location when the PREF command is issued.
MARK <i>setname</i>	Specifies a 1 to 10 character SBCS or a 1 to 4 character DBCS name given to a set using the COPY, MARK, MERGE, or RENAME commands.
PERFRANGE <i>name</i>	Specifies the first statement within the specified PERFORM range.
PROGRAM <i>name</i>	Specifies the first executable statement within the specified program.

Operand	Description																																							
SUBSET <i>name</i>	<p>Specifies a predefined subset. These are the valid subsets:</p> <table border="0"> <tr> <td>ASsignment</td> <td>DEBug</td> <td>IO</td> </tr> <tr> <td>CAll</td> <td>DEFinition</td> <td>LABel</td> </tr> <tr> <td>CIcs</td> <td>DIRective</td> <td>MAINline</td> </tr> <tr> <td>COBOLII</td> <td>DIVision</td> <td>MATH</td> </tr> <tr> <td>COBOL/370</td> <td>DL/I   DL/1</td> <td>Output</td> </tr> <tr> <td>COMment</td> <td>DML</td> <td>PARagraph</td> </tr> <tr> <td>CONditional</td> <td>ENtry</td> <td>PERform</td> </tr> <tr> <td>COPy</td> <td>EXIt   PGM</td> <td>ExitRETurn</td> </tr> <tr> <td>DB2/SQL</td> <td>FALLthrough</td> <td>SECTion</td> </tr> <tr> <td>DDL</td> <td>GOto</td> <td>SORTMerge</td> </tr> <tr> <td>DEAD</td> <td>IDMS</td> <td>STructure</td> </tr> <tr> <td>DEADCode</td> <td>INClude</td> <td>TESTed</td> </tr> <tr> <td>DEADData</td> <td>Input</td> <td>UNTested</td> </tr> </table> <p>A screen subset:</p> <p>Highlighted   HI  NONHighlighted   NHI  Excluded   X  NONExcluded   NX</p> <p>Tagged lines subsets of tags in columns 73 through 80.</p> <p><b>Note:</b> _____  See the <i>ASG-Insight User's Guide</i> for a description of each subset</p>	ASsignment	DEBug	IO	CAll	DEFinition	LABel	CIcs	DIRective	MAINline	COBOLII	DIVision	MATH	COBOL/370	DL/I   DL/1	Output	COMment	DML	PARagraph	CONditional	ENtry	PERform	COPy	EXIt   PGM	ExitRETurn	DB2/SQL	FALLthrough	SECTion	DDL	GOto	SORTMerge	DEAD	IDMS	STructure	DEADCode	INClude	TESTed	DEADData	Input	UNTested
ASsignment	DEBug	IO																																						
CAll	DEFinition	LABel																																						
CIcs	DIRective	MAINline																																						
COBOLII	DIVision	MATH																																						
COBOL/370	DL/I   DL/1	Output																																						
COMment	DML	PARagraph																																						
CONditional	ENtry	PERform																																						
COPy	EXIt   PGM	ExitRETurn																																						
DB2/SQL	FALLthrough	SECTion																																						
DDL	GOto	SORTMerge																																						
DEAD	IDMS	STructure																																						
DEADCode	INClude	TESTed																																						
DEADData	Input	UNTested																																						
LINE <i>range</i>	Specifies a single line number or range of lines. Line numbers display in columns 1 through 6 of Source View.																																							
LABEL <i>name</i>	Specifies any PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. All transfers of control to the label name are included.																																							
PARAGRAPH <i>name</i>	Specifies a PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. The entire paragraph or section is included.																																							

Operand	Description
Next	Generates a list of paragraphs to where the target paragraph transfers control.
Prev	Generates a list of paragraphs that transfer control to the target paragraph. This is the default for the PREF command.

### Usage Notes

You can also display the View - Paragraph Cross Reference pop-up by selecting View ► Paragraph X-ref and pressing Enter.

The PREF command includes all paragraphs containing the target lines. For example, PREF IO displays every paragraph containing IO statements.

To concatenate targets, place a plus sign (+) between the target names. For example, IO+CALL. A concatenated set name can be used wherever a set name is valid. These sets can be concatenated:

- LABEL
- MARK
- PERFRANGE
- SUBSET
- LINE
- PROGRAM
- PARAGRAPH

For COBOL II Release 3 or later programs, you can qualify a data item, label, or program name that may be ambiguous or used multiple times by using OF followed by the program name. For example, if the label P120-READ exists in another program and also in the program VIAIDEM1, then it can be qualified as P120-READ OF VIAIDEM1.

When there is a blank target with the cursor located on excluded lines, these items apply:

- The first line of the excluded block is implicitly selected.
- If the selected line is not a label, then a backward search is done to locate the previous label. This label then becomes the target.

**Note:** \_\_\_\_\_

See the online help for more information about the View - Paragraph Cross Reference pop-up.

\_\_\_\_\_

### Example

The command below displays the View - Paragraph Cross Reference pop-up with all paragraphs that transfer control to the WRITE-ROUTINE paragraph.

```
PREF WRITE-ROUTINE
```

## PRINTLOG Command

PRINTLOG | PLOg

---

### *Function*

The PRINTLOG command is used to display the Options - Log/List/Punch Definition pop-up. On this pop-up the Log, List, and Punch files are processed.

### *Operands*

None.

### *Usage Notes*

The Options - Log/List/Punch Definition pop-up can also be displayed by selecting Options ► Log/list/punch and pressing Enter.

The PRINTLOG command can be issued on any Insight screen.

## PRINTLST Command

PRINTLST | PList

---

### *Function*

The PRINTLST command is used to display the Options - Log/List/Punch Definition pop-up. On this pop-up the Log, List, and Punch files can be printed.

### *Operands*

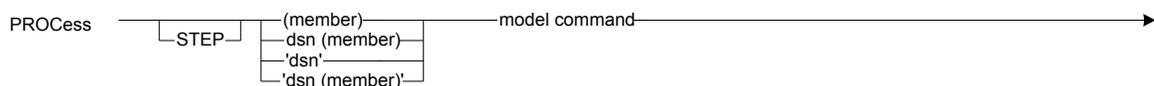
None.

### *Usage Notes*

The Options - Log/List/Punch Definition pop-up can also be displayed by selecting Options ► Log/list/punch and pressing Enter.

The PRINTLST command can be issued from any Insight screen.

## PROCESS Command



### Function

The PROCESS command defines an action to be performed repetitively on a file of input items.

This file may contain the results list from an ESW product, or the file can be nested by the user. The input items, called tokens, may be any text without embedded blanks or any text enclosed in quotes. Tokens must be separated by one or more spaces.

The PROCESS primary command process the first nine tokens in a file. Any tokens beyond nine are ignored. Any tokens referenced in the PROCESS command but not present in the token file are passed as null.

Tokens enclosed in quotes retain their quotes after substitution in the model statement.

This example illustrates a possible format for a record in the token file:

```
token1 'token 2' token3 token4 token5 'token 6' token7 token8 token9
```

Insight, Encore, SmartTest, and Alliance can produce an output file using LPRINT, LPUNCH, or the Query Facility. This file can be edited to produce a file of tokens that can be read by the PROCESS primary command.

### Operands

Operand	Description
STEP	Steps through the commands in the script file. Each command in the script file displays in the command input area. The displayed command can be changed or erased from the command input area. Press Enter to execute the displayed command. Processing of the script file continues in this manner until all commands have been displayed or executed.
( <i>member</i> )	Specifies a member in the default script library (defined during product installation) that contains tokens to be processed. The default script libraries may be modified for your profile using the Script file allocation action on the Options pull-down. Parentheses are required if only the member is entered.

Operand	Description
<i>dsn(member)</i>	<p>Specifies a partitioned dataset member that contains tokens to be processed. This dataset must be in card image format (LRECL=80). The specified dataset name and member must be entered following TSO usage conventions in this format:</p> <pre>'pds.dataset.name(member)'</pre> <p>Quotes are required unless the high-level qualifier is your TSO UserID.</p>
<i>dsn</i>	<p>Specifies a sequential dataset that contains tokens to be processed. This dataset must be in card image format (LRECL=80). The specified dataset must be entered following TSO usage conventions in this format:</p> <pre>'sequential.dataset.name'</pre> <p>Quotes are required unless the high-level qualifier is your TSO prefix or userID.</p>
Model command	<p>Specifies any ESW product command that is valid in a Script file and that can be defined using substitution variables. Tokens from the file are always substituted in the model command in numerical order from 1 through 9 for the substitution variables &amp;1 through &amp;9.</p> <p>When the PROCESS command is included in a script, the model command may optionally reference substitution variables that were passed to the script. This is done by specifying any of the override substitution variables &amp;&amp;1 through &amp;&amp;9.</p>

### Usage Notes

The PROCESS command allows automated processing of the results from any ESW product that can produce a text file of tokens.

For example, if Alliance creates a file of records named *userid.PROGDATA.TEXT* where each record contained a program name and a dataname, the file can be processed with the command:

```
PROCESS PROGDATA.TEXT EXECUTE SCANPROG.SCRIPT PARS &1 &2
```

This command reads each record of the file *PROGDATA.TEXT* in turn and substitutes program name from the record into substitution variable &1 and dataname from the record into substitution variable &2. It then executes the named script (*userid.SCANPROG.SCRIPT*), passing the substituted variables &1 and &2.

The script file SCANPROG.SCRIPT might contain these commands to qualify a program for viewing and execute another script file for the program.

```
Q &1  
EXECUTE PROGPROC.SCRIPT PARMS &2
```

This allows the processing of multiple programs with a single command sequence.

To accomplish this task, the PROCESS command verifies and opens the specified file. These logic steps are repeated until an end-of-file occurs for the specified file.

### ***To process multiple programs with a single command sequence***

- 1** Read a record. If the record is null or contains an asterisk (\*) in column 1, ignore the record and begin again.
- 2** Substitute tokens from the record into the substitution variables &1 through &9.
- 3** Construct a command using the model statement with the variable substitution.
- 4** Execute the constructed command as if it were a script command.
- 5** Repeat [step 1](#) and subsequent steps until you reach an end-of-file.

PROCESS commands may occur within scripts and may be nested. If PROCESS commands are nested, they must refer to different datasets.

## **Examples**

These examples show how you might use the PROCESS command in its most basic form to process a dataname list and then how this basic concept could be extended to automate the same action for a list of programs.

The third example illustrates the use of override substitution variables.

**Example 1.** Given a file of datanames called *userid.DATANAME.LIST*, you could execute this PROCESS command:

```
PROCESS DATANAME.LIST HI &1
```

Executing this command while on the Program View screen causes the HIGHLIGHT command (HI) to be executed repetitively for each data item in the file *userid.DATANAME.LIST*.

**Example 2.** Given a file of program names called *userid.PROGRAM.LIST*, you could execute this PROCESS command:

```
PROCESS PROGRAM.LIST EXEC MYSCRIPT.SCRIPT PARMS &1
```

Assume that *userid*.MYSCRIPT.SCRIPT contains these commands:

```
* STEP 1: OPEN THE PROGRAM
QUALIFY &1
* STEP 2: POSITION TO THE PROGRAM VIEW SCREEN (SmartTest only)
FX 1
* STEP 3: PROCESS THE DATA NAME LIST
PROCESS DATANAME.LIST HI &1
* STEP 4: SAVE THE RESULTS OF THE PROCESS COMMAND
SAVE MARKS AS &1MK
* STEP 5: CLOSE THE PROGRAM
QUALIFY CANCEL
```

The set of commands in *userid*.MYSCRIPT.SCRIPT execute one time for each program in *userid*.PROGRAM.LIST. *userid*.MYSCRIPT.SCRIPT contains a PROCESS command that executes the model command, HIGHLIGHT, one time for each record in *userid*.DATANAME.LIST.

The result is a MARK set, named *pgmname*MK, for each program in *userid*.PROGRAM.LIST. Each MARK set contains highlighted references to uses in the program of all datanames in *userid*.DATANAME.LIST.

**Example 3.** This example illustrates the technique of passing override substitution variables &&1 and &&2 to the PROCESS command contained in the script.

```
EXEC (MYSCRIPT) PARS ARG1 ARG2 ARG3
```

Assume the script in member MYSCRIPT in the default script libraries contains this PROCESS command:

```
PROCESS 'PROGRAM.LIST' EXEC (PGSCRIPT) PARS &&2 &1 &2 &&1
```

The parameters from the initial EXECUTE statement and the tokens from the file are substituted in the PROCESS statement model command:

```
&&2 is passed the value ARG2
&1 is passed the first token of each record read from the file
'PROGRAM.LIST'
&2 is passed the second token of each record read from the
file 'PROGRAM.LIST'
&&1 is passed the value ARG1
```

## PRODLVL Command

PRODLVL 

### Function

The PRODLVL command is used to display the current Insight and Center product level.

### Operands

None.

### Usage Notes

This information is requested when you contact the ASG Service Desk for assistance. This information can also be displayed from the Help - About pop-up. To display the Help - About pop-up, select Help ► About and press Enter.

The PRODLVL command displays the product name, operating system, product release number, and release level on the message line, in this format:

```
ASG1554I INSIGHT-OS (XA) Rn.n AT Lnnn, CENTER Rn.n AT Lnnn
```

where:

*Rn.n* is the release number.

*Lnnn* is the release level.

## PROGRAM Command



### Function

The PROGRAM command is used to end the current Source View session and enter a new Source View session for the specified program. This is useful when you are viewing a system of multiple programs.

### Operands

Operand	Description
Blank	Displays the AKR Program Selection pop-up for selection of a program.
<i>pgm</i>	Specifies a COBOL program or an entry point name. If the specified program or entry point name is unavailable in the current AKR, an error message displays.
Save	Saves equates and marks for this program in the Application Knowledge Repository. This operand overrides the values on the Options - Product Parameters pop-up.
Nosave	Specifies that equates and marks for this program are not saved. This operand overrides the values on the Options - Product Parameters pop-up.

### Usage Notes

You can also enter a new Source View session for a new program from the File - Open Program Request pop-up. To display the File - Open Program Request pop-up, select File ► Open and press Enter.

Using the PROGRAM command is the easiest way of switching to another program. However, Insight does not save the calling sequence. This means the GOBACK command cannot be issued to return to a previous program.

### Example

When current Source View program is VIAIDEMO and this command is entered:

```
PROGRAM VIAIDEM1 NOSAVE
```

The VIAIDEM1 program displays without saving marks and equates for VIAIDEMO.

## QUALIFY Command



### Function

The QUALIFY command displays the specified program on the Source View screen for investigation. The qualified program becomes the current program, and program understanding commands operate on the qualified program. Additional programs can be qualified as needed.

### Operands

Operand	Description
<i>pgm</i>	Specifies the program to be displayed on the Source View screen.
CANcel	Closes the current program.
ALL	Closes all open programs.
<i>AKRname</i> <i>pgm</i>	Specifies an AKR name. If you do not specify another AKR name, Insight uses the profile AKR in batch.

## RECALL Command



### Function

The RECALL command is used to display the previous primary command, message, or pop-up. The last 20 commands executed and the last 20 messages displayed are stacked. Use the RECALL command to redisplay these commands or messages.

### Operands

Operand	Description
Blank	Displays the last primary command that was stacked. After you enter the RECALL command operands, subsequently typing RECALL with no operands reuses the same operands that were last entered.
COMmand   CMD	Displays a stacked primary command. This is the default value.
MESsage   MSG	Displays a stacked message.
NEXT	Displays the next command or message in the stack.
PREV	Displays the previous command or message in the stack. This is the default value.
POPup	Displays the pop-up that was most recently requested from a pull-down.

### Usage Notes

You can enter the RECALL command repeatedly to display any of the 20 stacked commands, messages, or pop-ups. The NEXT and PREV operands can be used to move forward or backward through the stacked commands or messages. After the desired command displays, you can execute it again by pressing Enter. Any command that is recalled can be changed before you execute it. To re-execute the last saved primary command again without modification, use the REPEAT command.

The operands you specify for the RECALL command remain in effect until one of these conditions occur:

- A different operand is specified.
- A different primary command is executed. When this occurs, the RECALL command default operands are automatically set. A message displays that indicates all stacked commands or messages have been shown and the stack is being redisplayed.

These commands are not stacked for use by the RECALL command:

ANALYZE	LOCATE	RECALL	RSCROLL
CALL	PARMDEF	REDO	RTRACE
END	PRINTLOG	REPEAT	RTREEVW
GOBACK	PRINTLST	RETURN	SAVE
HELP	PRODLVL	RFIND	UPDATE
KEYS	PROGRAM	RHIGH	VIEW

### *Example*

This example assumes these seven commands were entered for the current program and that seven messages were displayed:

```
1  X ALL
2  ZOOMIN
3  PREF
4  ZOOMOUT
5  FX
6  SCROLL
7  FX
```

To display the previous message (number 7), type `RECALL MSG`. The `MSG` operand remains in effect until the `COMMAND` or `CMD` operand is specified.

Therefore, typing `RECALL` without changing the operands displays the previous message (number 6). For example, typing `RECALL` with no operands displays the previous command (number 7). The `CMD` operand remains in effect until a `MSG` operand is specified.

```
RECALL CMD
```

Therefore, typing `RECALL` without changing the operands displays the previous command (number 6).

This command displays the next command (number 7).

```
RECALL MD NEXT
```

## REDO Command



### Function

The REDO command executes the corresponding repeat command from the cursor position, as shown below.

Last Command	Command Executed
FIND	RFIND
FINDXTND	RFIND
HIGH	RHIGH
PREF	RPREF
SCROLL	RSCROLL
TRACE	RTRACE

Only the commands listed in the table above are re-executed when you type REDO.

### Operands

Operand	Description
Blank	Re-executes the last command entered. These operands are valid only if the last command was TRACE. Otherwise, they are ignored.
<i>trace-dec-opt</i>	Specifies a TRACE command decision option.
BACKup	Returns to the decision point where the last decision option was entered.

### Usage Notes

REDO automatically executes the corresponding repeat command, depending on the last command entered. For example, if the last command entered was FIND, RFIND is executed.

## REFRESH Command

REFresh 

### *Function*

Updates the call summaries for programs called by the active program.

### *Operands*

None.

### *Usage Notes*

Call summaries can also be updated by selecting Refresh on the Options pull-down. See the online help for more information.

REFRESH is used to update information in the active program regarding any CALLED program whose source was analyzed since opening the program.

## RENAME Command



### Function

The RENAME command changes the name of a mark path or set of lines.

### Operands

Operand	Description
<i>mark-name</i>	Specifies a name assigned to a path or set of lines using the MARK command or the Options - Scratchpad Mark pop-up. The specified name must meet these requirements: <ul style="list-style-type: none"> <li>• Must be a maximum of 10 alphanumeric or 4 DBCS characters and can include hyphens.</li> <li>• Names longer than 10 characters truncate.</li> <li>• Name must begin with an alphabetic character.</li> </ul>
TO	Specifies an optional keyword indicating the name following is the new mark name.
NOTE comment	Includes an optional description about the mark name. Comment text can be a maximum of 50 alphanumeric or 23 DBCS characters. If the NOTE comment operand is not entered, the existing comment is copied to the new mark name. If NOTE is entered without comment text, the existing comment is not copied to the new mark name.

### Usage Notes

Marks can also be renamed on the Options - Scratchpad Rename pop-up. See the online help for more information.

An error message displays if the mark name to be renamed does not exist or the resulting mark name already exists.

### Example

When this command is entered, the mark name SETA is changed to SETB. SETA is no longer valid as a mark name.

```
RENAME SETA TO SETB
```

## REPEAT Command

REPEat 

### Function

Executes the last stacked primary command again.

### Operands

None.

### Usage Notes

Type REPEAT in the command input area to re-execute the last stacked primary command. The command is executed from the cursor position.

To modify the command before it is re-executed, use the RECALL command.

### Example

To display the next occurrence of a highlighted line, type this command:

```
SCROLL HI NEXT
```

Instead of re-entering the same command, type this command:

```
REPEAT
```

This command executes the SCROLL HI NEXT command again from the cursor position.

## RESET Command



### Function

The RESET command removes highlighting and tags, cancels pending line commands, terminates message lines, and/or redisplay all excluded lines.

### Operands

Operand	Description
Blank	Defaults to the value ALL.
All	Resets all conditions indicated by each of the operands. This is the default.
Hi	Removes the highlighting from any line that was highlighted as a result of another Insight command.
Tag	Erases tags in columns 73 through 80 set as a result of another Insight command.
Excluded   X	Displays any excluded lines.
Lines	Clears any pending line commands.
Message	Erases short or long messages that display.
<i>Line range</i>	Resets the HI, TAG, EXCLUDED, LINES, and MESSAGE conditions only for the specified line or range of lines.
MARK <i>markname</i>	Resets the HI, TAG, EXCLUDED, LINES, and MESSAGE conditions only for statements belonging to the mark set. The keyword MARK is not needed if there is no ambiguity.

### Usage Notes

You can also reset Features from the View - Reset pop-up. To display the View - Reset pop-up, select View ► Reset and press Enter. Any combination of operands can be specified for the RESET command.

## RETURN Command



### Function

The RETURN command is used to terminate the current function and return to the first Insight screen displayed.

### Operands

Operand	Description
Blank	Specifies that the values specified on the Options - Product Parameters pop-up are used to determine if marks and equates are to be saved.
Save	Saves marks and equates created for this program in the AKR. This operand overrides the values set on the Options - Product Parameters pop-up.
Nosave	Specifies that marks and equates are not saved for this program. This operand overrides the values set on the Options - Product Parameters pop-up.

### Usage Notes

Using the RETURN command is a quick way of ending an Insight function.

## RFind Command

RFind 

### Function

The RFind command is used to repeat the last FIND or FINDXTND command from the cursor position.

### Operands

None.

### Usage Notes

The RFind command is a convenient way to locate the next occurrence of the specified target of a FIND or FINDXTND command. The search is performed in the direction indicated in the FIND or FINDXTND command. RFind is assigned to the PF05/17 key as a default.

The REDO command can also be used to execute the RFind command. See ["REDO Command" on page 286](#) for more information.

## RHIGH Command

RHgh 

### *Function*

The RHIGH command is used to repeat the last HIGH command from the cursor position.

### *Operands*

None.

### *Usage Notes*

Type RHIGH in the command input area to repeat the last HIGH command from the cursor position.

The REDO command can also be used to execute the RHIGH command. See ["REDO Command" on page 286](#) for more information.

## RPREF Command

RPref 

### Function

When on the Source View screen, use the RPREF command to redisplay the last View - Paragraph Cross Reference pop-up. When on the View - Paragraph Cross Reference pop-up, use the RPREF to redisplay the last Source View screen.

### Operands

None.

### Usage Notes

The View - Paragraph Cross Reference pop-up redisplay with the exact information as previously shown. RPREF can be used to go back and forth between the source and the View -Paragraph Cross Reference pop-up.

When an RPREF (and no PREF) command is entered, RPREF functions as a PREF command with the cursor position as the target and default direction.

The REDO command can also be used to execute the RPREF command. See ["REDO Command" on page 286](#) for more information.

## RSCROLL Command

RSCroll 

### Function

The RSCROLL command is used to repeat the last SCROLL command from the cursor position.

### Operands

None.

### Usage Notes

Type RSCROLL in the command input area to repeat the last SCROLL command from the cursor position.

The REDO command can also be used to execute the RSCROLL command. See ["REDO Command" on page 286](#) for more information.

## RSTRUCTURE-VIEW Command

RSTRUCTure-view | RSTView

---

### *Function*

The RSTRUCTURE-VIEW command is used to return to the previous Perform Structure View screen.

### *Operands*

None.

## RTRACE Command



### Function

The RTRACE command is used to continue the last TRACE command from where it stopped.

### Operands

Operand	Description
Blank	Displays the Trace Decision Options pop-up. When RTRACE is entered without an operand and not at a decision point, the trace function is resumed from the stopping point.
<i>trace-dec-opt</i>	Specifies a TRACE command decision option.
BACKUp	Returns to the decision point where the last decision option was entered. The TRACK is reset to include only those lines that were in it when that decision point was reached. Only one BACKUP is allowed following an RTRACE command. A message displays if there is no previous decision point or if BACKUP has already been done.

### Usage Notes

The RTRACE command can be used to continue a trace function without exiting the Source View screen. Type the desired option number in the command input area and press Enter. This process can be continued until the end of the trace function is reached. If a different path is to be followed, type BACKUP in the command input area, press Enter, and select the desired option.

If a trace function is interrupted and another primary command is entered, the messages displayed by the trace function are cleared. To continue the trace function, type RTRACE. The screen is then scrolled to the last trace location and the messages redisplay.

If a decision point prompt is unclear, the Trace Decision Options pop-up that explains the decision point can be displayed. Type RTRACE without an operand to display the Trace Decision Options pop-up. The Trace Decision Options pop-up shows this information: the possible options from the decision point; the type of branch; the line number; and the first statement of the destination. An asterisk (\*) next to an option indicates that option was traced.

The REDO command can also be used to execute the RTRACE command. See "[REDO Command](#)" on page 286 for more information.

### Trace Decision Options Pop-up

The Trace Decision Options Pop-up ([Figure 102 on page 299](#)) displays when you type RTRACE without an operand. This pop-up is used to list all possible options from the decision point, the type of branch, the line number, and the first statement of the destination. An asterisk displayed next to an option shows that you have already TRACEd that option.

Figure 102 • Trace Decision Options Pop-up

```

                                Trace Decision Options
Command ==> _____ Scroll ==> CSR

Trace of: TESTCALL
Direction: FORWARD
Paragraph: 000015 PROCEDURE DIVISION.
Decision: 000018 IF WS-CHKP-COUNTER = 10

Option Info indicates TARGET's and '*' for code already TRACE'd

Select one of the following decision options:
$ Option                                     Option
-----                                     -----
_ 1. TRUE (000019 MOVE 0 TO WS-CHKP-INTERVAL,WS-CHKP-COUNTER)
_ 2. FALSE (000020 ELSE)
***** BOTTOM OF DATA *****
    
```

### Fields

Field	Description
Trace of	Specifies the name of the target you are tracing. For a dataname, it also indicates the usage (USE, MOD, REF). If an IN clause was specified, it is also displayed.
Direction	Specifies the direction requested for the trace (NEXT, PREVIOUS, FORWARD, or BACKWARD).
Paragraph	Specifies the line number and name of the paragraph that contains the decision point.
Decision	Specifies the line number and statement of the decision point.
S (line command area)	Specifies the line command area, to the left of the Option field, accepts the S line command to select the option.

Field	Description
Option	Specifies the all feasible options from the decision point. Each option lists the type of branch, line number, and first statement of the destination.
Option Info	Indicates whether the option is the requested target or has already been traced.

## RTREEVW Command

RTREEw | RTV



### *Function*

The RTREEVW command returns to the previous Tree View screen.

### *Operands*

None.

### *Usage Notes*

The Tree View screen is redisplayed with the exact information as previously shown. See the online help for more information about the Tree View screen.

## SAVE Command



### Function

The SAVE command is used to save marks and/or equates in the AKR. This command overrides the values set on the Options - Product Parameters pop-up.

### Operands

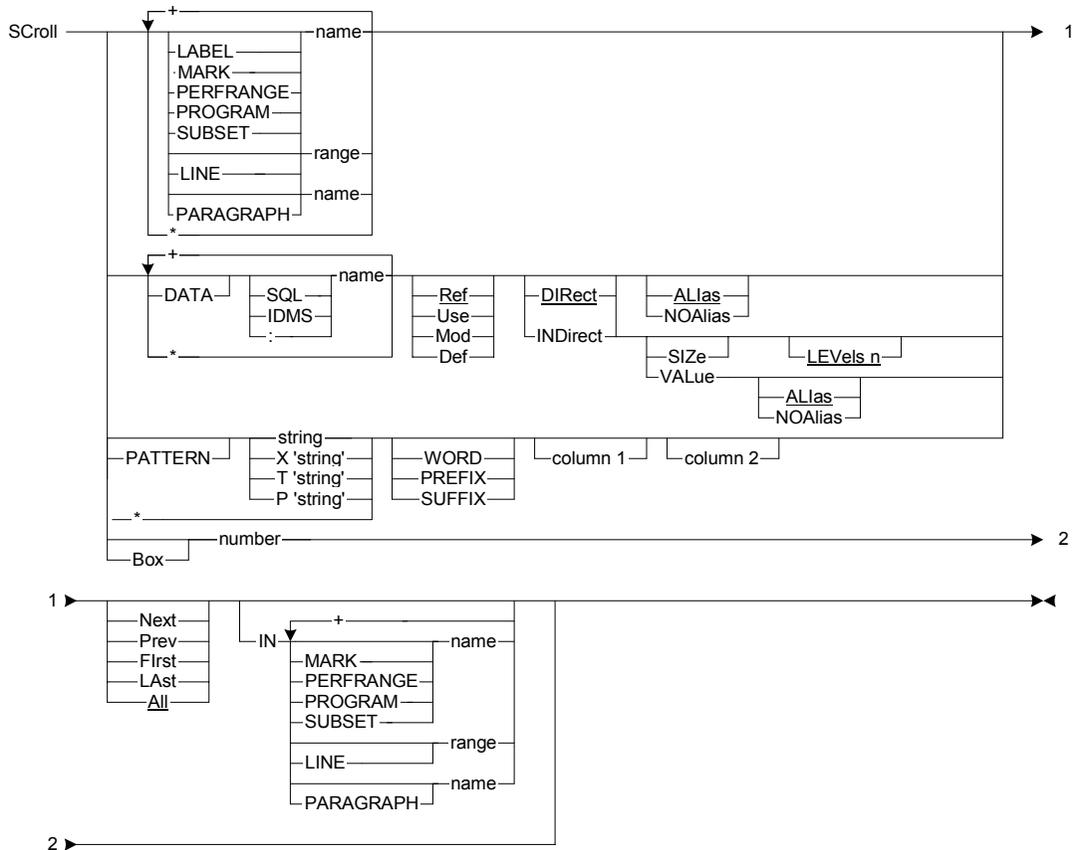
Operand	Description
Blank	Defaults to the value ALL.
Marks	Saves all marks created or changed for the active program. This operand overrides the values set on the Options - Product Parameters pop-up.
Equates	Saves all equates entered for the active program. This operand overrides the values set on the Options - Product Parameters pop-up.
All	Saves marks and equates entered for the active program. This operand overrides the values set on the Options - Product Parameters pop-up. This is the default.

### Usage Notes

Marks and equates can also be saved on the File - Save pop-up, and on the File - Close pop-up. See the online help for more information.

Save values established on the Options - Product Parameters pop-up can be overridden using the SAVE command.

## SCROLL Command



### Function

The SCROLL command is used to position the screen to the first line containing the specified target. Highlighted lines remain unchanged.

### Operands

Operand	Description
LABEL <i>name</i>	Specifies the paragraph or section name of the PROCEDURE DIVISION or the literals PROCEDURE or PROC where the FLOW command is to begin.

Operand	Description																					
MARK <i>name</i>	<p>Specifies a 1 to 10 alphanumeric character name given to a set or path using the COPY, MARK, MERGE, or RENAME commands, or one of these system-generated paths:</p> <table border="0"> <tr> <td>System generated path:</td> <td>Created by:</td> </tr> <tr> <td>TRACK   TRK</td> <td>A TRACE command.</td> </tr> <tr> <td>NETWORK   NET</td> <td>A FLOW command.</td> </tr> <tr> <td>SUBNET<sub>n</sub>   SUB<sub>n</sub></td> <td>A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (n<sup>th</sup> result).</td> </tr> </table>	System generated path:	Created by:	TRACK   TRK	A TRACE command.	NETWORK   NET	A FLOW command.	SUBNET <sub>n</sub>   SUB <sub>n</sub>	A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (n <sup>th</sup> result).													
System generated path:	Created by:																					
TRACK   TRK	A TRACE command.																					
NETWORK   NET	A FLOW command.																					
SUBNET <sub>n</sub>   SUB <sub>n</sub>	A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (n <sup>th</sup> result).																					
<p><b>Note:</b></p> <p>The TRACK, NETWORK, and SUBNET paths can also be created using the Logic Order Search pop-ups (see <a href="#">"Logic Pull-down" on page 101</a>).</p>																						
PERFRANGE <i>name</i>	Specifies the name in a PERFORM statement including all statements executed as a result of a PERFORM statement, or the name of any section contained in the Declaratives. This operand is valid in Source View only																					
PROGRAM <i>name</i>	Specifies the name of the main program or any nested program representing all the code contained in the program. This includes all the programs physically nested inside the specified program.																					
SUBSET <i>name</i>	<p>Specifies a predefined subset. These are the valid subsets:</p> <table border="0"> <tr> <td>ASsignment</td> <td>DEBug</td> <td>IO</td> </tr> <tr> <td>CALL</td> <td>DEFinition</td> <td>LABel</td> </tr> <tr> <td>CIes</td> <td>DIRective</td> <td>MAINline</td> </tr> <tr> <td>COBOLII</td> <td>DIVision</td> <td>MATH</td> </tr> <tr> <td>COBOL/370</td> <td>DL/I   DL/1</td> <td>Output</td> </tr> <tr> <td>COMment</td> <td>DML</td> <td>PARagraph</td> </tr> <tr> <td>CONditional</td> <td>ENtry</td> <td>PERform</td> </tr> </table>	ASsignment	DEBug	IO	CALL	DEFinition	LABel	CIes	DIRective	MAINline	COBOLII	DIVision	MATH	COBOL/370	DL/I   DL/1	Output	COMment	DML	PARagraph	CONditional	ENtry	PERform
ASsignment	DEBug	IO																				
CALL	DEFinition	LABel																				
CIes	DIRective	MAINline																				
COBOLII	DIVision	MATH																				
COBOL/370	DL/I   DL/1	Output																				
COMment	DML	PARagraph																				
CONditional	ENtry	PERform																				

Operand	Description
	COPy          EXIt   PGM    ExitRETurn
	DB2/SQL      FALLthrough   SEctIon
	DDL          Goto          SORTMerge
	DEAD        IDMS         SStructure
	DEADCode    INclude      TESTed
	DEADData    Input         UNTested
	A screen subset:
	Highlighted   HI
	NONHighlighted   NHI
	Excluded   X
	NONExcluded   NX
	Tagged lines subsets of tags in columns 73 through 80.
	<b>Note:</b> _____ See the <i>ASG-Insight User's Guide</i> for a description of each subset _____
LINE <i>range</i>	Specifies a single line number or range of lines. Line numbers display in columns 1 through 6 of Source View.
PARAGRAPH <i>name</i>	Specifies a PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. The entire paragraph or section is included.
asterisk (*)	Reuses the target of the previous SCROLL.
DATA <i>name</i>	Specifies a COBOL dataname or qualified COBOL dataname. Dataname refers to any valid COBOL reference for a data element.  These subordinate operands apply to the dataname and can be used to further qualify the EXCLUDE command: REF, USE, MOD, DEF, ALIAS, NOALIAS, DIRECT, and INDIRECT. The defaults are REF, ALIAS, and DIRECT.
SQL <i>name</i>	Includes datanames that are DB2/SQL variables only.
IDMS <i>name</i>	Includes datanames that are IDMS variables only.
: <i>name</i>	Includes datanames that are COBOL variables only.

Operand	Description
Ref	Includes occurrences of the dataname where its value is defined, tested, used, set, or modified. This is the default value for the DATA name operand.
Use	Excludes occurrences of the dataname where its value is being tested or used.
Mod	Excludes occurrences of the dataname where its value is being set or modified.
Def	Includes definitions of the dataname in the DATA DIVISION
ALias	<p>Searches backward from the cursor position to the previous occurrence of the requested target.</p> <p>Includes occurrences of the aliases for the dataname. These are the valid aliases:</p> <ul style="list-style-type: none"> <li>• Parent - higher level group item</li> <li>• Child - lower level item</li> <li>• Rename/Redefinition - renamed, redefined, or 88 level items</li> </ul> <p>This is the default value for the DATA name operand.</p>
NOAlias	Ignores aliases for the specified dataname.
DIRect	Includes occurrences of the dataname (and aliases if specified) where it is directly tested, used, set, modified, or defined (based on the REF, USE, MOD, or DEF selection). The default for the DATA name operand is direct; however, if the SIZE, VALUE, or LEVELS operand is specified, INDIRECT is assumed.
INDirect	Includes any dataname indirectly affected by the specified dataname (and aliases if specified). The indirect data item can be further qualified using the SIZE, VALUE, and LEVELS subordinate operands. These subordinate operands indicate the type of indirect reference to be located. SIZE and all levels are the defaults for the INDIRECT operand.
SIZE	Includes datanames that could be directly or indirectly affected if the size of the specified dataname were to be changed. SIZE is only valid with the INDIRECT operand. This is the default for the INDIRECT operand.

Operand	Description
VALue	Includes occurrences of datanames that could be directly or indirectly affected by a change if the value of the specified dataname were to be changed. The LEVELS subordinate operand is not used with VALUE. VALUE is only valid with the INDIRECT operand.
LEVels <i>n</i>	Includes all data items affected within the specified number of indirect levels. The VALUE subordinate operand is not used with LEVELS. All levels is the default for the LEVELS operand. LEVELS is only valid with the INDIRECT operand.
PATTERN	Indicates the characters that follow are part of a string.
<i>string</i>	Specifies a string of alphanumeric or DBCS characters. If the pattern string contains blanks, it must be enclosed in single or double quotes. The pattern string can be further qualified using the WORD, PREFIX, or SUFFIX subordinate operands. These subordinate operands describe how the pattern string is to be used:
<i>X'string'</i>	Specifies the hexadecimal string option. Specific unprintable characters may be specified by giving the EBCDIC hexadecimal value. The string must be enclosed in single or double quotes.
<i>T'string'</i>	Specifies the text string option. A character string may be entered regardless of upper or lowercase by using the text option. The string must be enclosed in single or double quotes.
<i>P'string'</i>	Specifies the picture string option. A string profile may be entered instead of exact characters. The string must be enclosed in single or double quotes. Insight defines nine special characters for use in picture strings. These special characters can be combined with other characters.
	These are the special characters:
	P'=' Any character
	P'~' Any nonblank character
	P'!' Any nondisplay character
	P'#' Any numeric character
	P'-' Any character
	P'@' Any character
	P'<' Any character
	P'>' Any character
	P'\$' Any character

Operand	Description
WORD	Specifies the pattern string preceded and followed by any non-alphanumeric character (except a hyphen).
PREFIX	Specifies a word that begins with the specified pattern string.
SUFFIX	Specifies a word that ends with the specified pattern string.
Column1	Specifies the column number where the search is to begin.
Column2	Specifies the column number where the search is to end.
Next	Searches forward from the current cursor position to the next occurrence of the requested target.
Prev	Searches backward from the current cursor position to the previous occurrence of the requested target.
FIRST	Searches from the top of the source file to the first occurrence of the requested target.
LAST	Searches backward from the bottom of the source file to the first occurrence of the requested target.
All	Searches for all occurrences of the requested target and displays the number found in the short/long message field. This is the default value for the HIGH command.
IN	Restricts the HIGH command to the specified target type.
Box Number	Identifies a scroll to a specific box number, rather than a line. This operand is only available in Perform Structure View.

### **Usage Notes**

The screen can also be scrolled by selecting a Search pop-up on the Search pull-down. See the online help for more information.

To concatenate targets, place a plus sign (+) between the target names. For example, IO + CALLS. These rules apply to concatenation:

- A concatenated dataname can be used wherever a dataname is valid. Dataname subordinate operands (REF, ALIAS, etc.) pertain to all datanames in a concatenated series.

- A concatenated set name can be used wherever a set name is valid. These sets can be concatenated:
  - LABEL
  - MARK
  - PERFRANGE
  - SUBSET
  - LINE
  - PROGRAM
  - PARAGRAPH

The asterisk (\*) operand can be concatenated once with any number of other operands (e.g., \* + IO + MYSET); however, the asterisk (\*) operand cannot be concatenated to itself (\* + \* is invalid). The \* operand may appear in any order in the concatenated list.

For COBOL II Release 3 or later programs, a data item, label, or program name that may be ambiguous or used multiple times can be qualified by using OF followed by the program name. For example, if the label P120-READ exists in another program and also in the program VIAIDEM1, then it can be qualified as P120-READ OF VIAIDEM1.

### *Example*

This command displays the first occurrence of a line containing the characters ZIP.

```
SCROLL ZIP
```

## SELECT Command



### Function

The SELECT command is used to select the desired option from a list. SELECT is used on the Trace Decision Options pop-up and the AKR Program Selection pop-up. SELECT followed by the desired option can be entered in the command input area. An S can be entered to the left of the displayed items to select a particular item on any screen with a selection field.

### Operands

Operand	Description
<i>trace-dec-opt</i>	Specifies a TRACE command decision option.
<i>pgm</i>	Specifies the complete program name from the AKR Program Selection pop-up.

**Note:**

See ["RTRACE Command" on page 298](#) for more information about the Trace Decision Options pop-up.

---

## SET Command



### Function

The SET command is used to enable or disable the mode indicated by the specified operand.

### Operands

Operand	Description
Blank	Displays the Options - Processing Modes pop-up that shows the current setting for each SET command mode.
LEarn	Displays all primary commands generated internally when actions from the pull-downs execute. The default for this operand is OFF.
SScript	Allocates a script file when the SET SCRIPT ON command is issued. From that point on, all Insight primary commands are captured in this file. When the SCRIPT OFF command is issued, the script file is closed and saved. The script file name is <i>userid.INSnnnnn.VIASCRIPT</i> . If the TSO profile prefix is not the same as the user ID, then this name is preceded by a prefix value.
RESULT	Saves the results of primary commands as comments in the script file when used with the SCRIPT operand.
XMode	Excludes all lines from the screen before a primary command is executed when this mode is ON. This mode affects all primary commands with the exception of these: <ul style="list-style-type: none"> <li>• BRANCH</li> <li>• LOCATE</li> <li>• SCROLL</li> <li>• HIGH</li> <li>• RFIND</li> </ul> <p>The default value for this operand is OFF.</p>

## Usage Notes

Any modes that you either enabled or disabled with the SET command can also be enabled or disabled on the Options - Processing Modes pop-up.

To display the Options - Processing Modes pop-up, follow this step:

- ▶ Select Options ▶ Modes and press Enter.

**Or**

Type SET (with no operands) on any screen and press Enter.

See the online help for more information about the Options - Processing Modes pop-up.

## LEARN Mode - Generated Command Pop-up

The LEARN Mode - Generated Command pop-up displays the internally-generated primary command when actions are requested on pop-ups, if the LEARN mode is set to ON with the SET command or on the Options - Processing Modes pop-up (shown in [Figure 103](#)).

**Figure 103 • LEARN Mode - Generated Command Pop-up**

```
LEARN Mode - Generated Command
To process the generated command, press Enter. To cancel
the command, press PF3/15 (END).
Command TREEVIEW 2
```

Follow the instructions on the pop-up to process or cancel the command.

## SOURCEVIEW Command

SOURCEVIEW | SView

---

### Function

The SOURCEVIEW command is used to display the Source View screen. This screen shows the program in source code order.

### Operands

None.



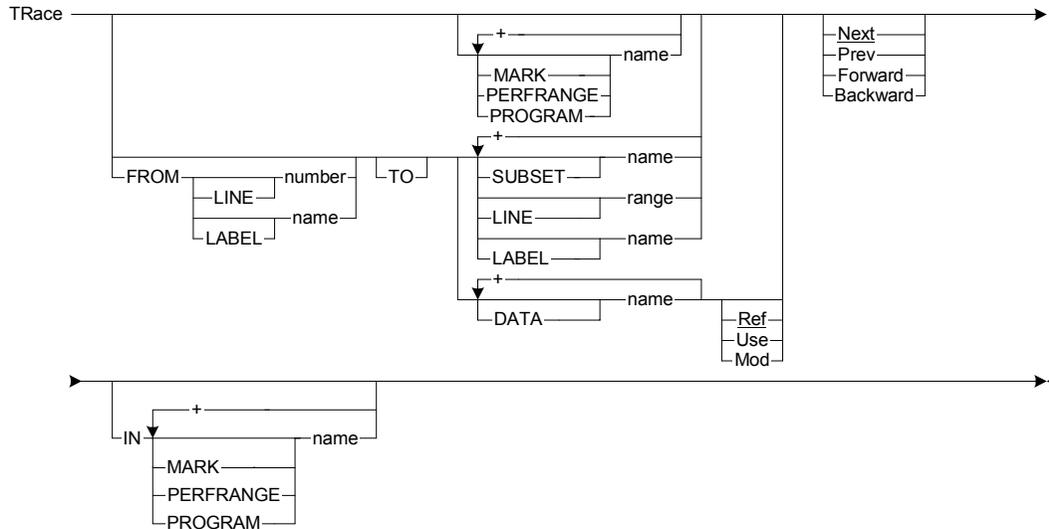
Operand	Description
LEVELs	Controls the amount of information displayed on entering Perform Structure View. Specify the number of nested control flow structures to be displayed. The default, 2, displays the first two levels of nested performs. MAX displays the maximum nesting level. However, if the maximum nesting level consists of only repeated perform ranges, then the nested level is truncated to the next higher level that does not consist exclusively of repeated perform ranges.
REPetition	Controls the display of duplicated perform range structures. If REPETITION is YES, all duplicate perform ranges display. If REPETITION is NO, duplicate perform range structures are only flagged as REPEATED. The default is NO.

### *Usage Notes*

The Perform Structure View screen can also be displayed by selecting View ► Structure and pressing Enter.

Type `STRUCTURE-VIEW` in the command input area to transfer the display to the Perform Structure View screen. Use the `END` command to return to screen where the `STRUCTURE-VIEW` command was entered.

## TRACE Command



### Function

The TRACE command is used to follow the execution of a program, searching for the specified target. The MARK and PERFRANGE operands trace the path represented by the MARK name, PERFRANGE name, or PROGRAM name. Tracing to a SUBSET name, LINE range, LABEL name, or DATA name traces from the starting point to the lines represented by these targets.

### Operands

Operand	Description
Blank	Specifies that the default direction of NEXT is used. The trace begins at the cursor position and continues to the program exit.
FROM	Indicates where the TRACE command is to begin. The MARK name, PERFRANGE name, and PROGRAM name operands cannot be used with this operand.
LINE <i>number</i>	Specifies the line number where the TRACE command is to begin when the target is a SUBSET name, LINE range, LABEL name, or DATA name.
LABEL <i>name</i>	Specifies the paragraph, section, or division name where the TRACE command is to begin when the target is a SUBSET name, LINE range, LABEL name, or DATA name.

Operand	Description																		
TO	Specifies an optional keyword that is followed by a target to indicate where the TRACE command is to end.																		
MARK <i>name</i>	<p>Specifies a 1 to 10 alphanumeric character name given to a set or path using the COPY, MARK, MERGE, or RENAME commands, or one of these system-generated paths:</p> <table border="0"> <tr> <td>System generated path:</td> <td>Created by:</td> </tr> <tr> <td>TRACK   TRK</td> <td>A TRACE command.</td> </tr> <tr> <td>NETWORK   NET</td> <td>A FLOW command.</td> </tr> <tr> <td>SUBNET<sub>n</sub>   SUB<sub>n</sub></td> <td>A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (n<sup>th</sup> result).</td> </tr> </table> <p><b>Note:</b> The TRACK, NETWORK, and SUBNET paths can also be created using the Logic Order Search pop-ups (see <a href="#">"Logic Pull-down" on page 101</a>).</p>	System generated path:	Created by:	TRACK   TRK	A TRACE command.	NETWORK   NET	A FLOW command.	SUBNET <sub>n</sub>   SUB <sub>n</sub>	A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (n <sup>th</sup> result).										
System generated path:	Created by:																		
TRACK   TRK	A TRACE command.																		
NETWORK   NET	A FLOW command.																		
SUBNET <sub>n</sub>   SUB <sub>n</sub>	A path created by the FLOW command that reaches from the beginning of the NETWORK to one of the results (n <sup>th</sup> result).																		
PERFRANGE <i>name</i>	Specifies the name in a PERFORM statement including all statements executed as a result of a PERFORM statement, or the name of any section contained in the Declaratives. This operand is valid in Source View only																		
PROGRAM <i>name</i>	Specifies the name of the main program or any nested program representing all the code contained in the program. This includes all the programs physically nested inside the specified program.																		
SUBSET <i>name</i>	<p>Specifies a predefined subset. These are the valid subsets:</p> <table border="0"> <tr> <td>ASsignment</td> <td>DEBug</td> <td>IO</td> </tr> <tr> <td>CALL</td> <td>DEFinition</td> <td>LABel</td> </tr> <tr> <td>CIcs</td> <td>DIRective</td> <td>MAINline</td> </tr> <tr> <td>COBOLII</td> <td>DIVision</td> <td>MATH</td> </tr> <tr> <td>COBOL/370</td> <td>DL/I   DL/1</td> <td>Output</td> </tr> <tr> <td>COMment</td> <td>DML</td> <td>PARagraph</td> </tr> </table>	ASsignment	DEBug	IO	CALL	DEFinition	LABel	CIcs	DIRective	MAINline	COBOLII	DIVision	MATH	COBOL/370	DL/I   DL/1	Output	COMment	DML	PARagraph
ASsignment	DEBug	IO																	
CALL	DEFinition	LABel																	
CIcs	DIRective	MAINline																	
COBOLII	DIVision	MATH																	
COBOL/370	DL/I   DL/1	Output																	
COMment	DML	PARagraph																	

Operand	Description
	<p>CONditional ENtry PERform</p> <p>COPy EXIt   PGM ExitRETurn</p> <p>DB2/SQL FALLthrough SECTion</p> <p>DDL GOto SORTMerge</p> <p>DEAD IDMS STRucture</p> <p>DEADCode INClude TESTed</p> <p>DEADData Input UNTested</p> <p>A screen subset:</p> <p>Highlighted   HI</p> <p>NONHighlighted   NHI</p> <p>Excluded   X</p> <p>NONExcluded   NX</p> <p>Tagged lines subsets of tags in columns 73 through 80.</p> <p><b>Note:</b> _____            See the <i>ASG-Insight User's Guide</i> for a description of each subset            _____</p>
LINE <i>range</i>	Specifies a single line number or range of lines. Line numbers display in columns 1 through 6 of Source View.
LABEL <i>name</i>	Specifies any PROCEDURE DIVISION paragraph name or section name, or the literals PROCEDURE and PROC. All transfers of control to the label name are included.
DATA <i>name</i>	<p>Specifies a COBOL dataname or qualified COBOL dataname. Dataname refers to any valid COBOL reference for a data element.</p> <p>These subordinate operands apply to the dataname and can be used to further qualify the EXCLUDE command: REF, USE, MOD, DEF, ALIAS, NOALIAS, DIRECT, and INDIRECT. The defaults are REF, ALIAS, and DIRECT.</p>
Ref	Includes occurrences of the dataname where its value is defined, tested, used, set, or modified. This is the default value for the DATA name operand.
Use	Excludes occurrences of the dataname where its value is being tested or used.
Mod	Excludes occurrences of the dataname where its value is being set or modified.

Operand	Description
Next	Searches forward from the current cursor position to the next occurrence of the requested target.
Prev	Searches backward from the current cursor position to the previous occurrence of the requested target.
Forward	Traces forward from the current cursor position to all occurrences
Backward	Traces backward from the current cursor position to all occurrences.
IN	Restricts the HIGH command to the specified target type.

### Usage Notes

Execution paths in a program can also be identified using the Logic Order Search pop-ups, that can create a path with a mark name of TRACK. See [Chapter 5, "Logic," on page 99](#) for more information.

To concatenate targets, place a plus sign (+) between the target names. For example, IO+CALL. These rules apply to concatenation:

- A concatenated dataname can be used wherever a dataname is valid. Dataname subordinate operands (REF, USE, MOD) pertain to all datanames in a concatenated series.
- A concatenated set name can be used wherever a set name is valid. These sets can be concatenated.
  - LABEL
  - MARK
  - PERFRANGE
  - LINE
  - PROGRAM
  - SUBSET

For COBOL II Release 3 or later programs, a data item, label, or program name that may be ambiguous or used multiple times can be qualified by using OF followed by the program name. For example, if the label P120-READ exists in another program and also in the program VIAIDEM1, then it can be qualified as P120-READ OF VIAIDEM1.

When TRACE is entered with only a direction operand, this occurs:

Direction	Traces to the...
Next	program exit statements.
Previous	PROCEDURE DIVISION statement or the program ENTRY points.

Direction	Traces to the...
Forward	program exit statements.
Backward	PROCEDURE DIVISION statement or the program ENTRY points.

The trace function can be performed on an existing path (e.g., NETWORK, SUB $n$ ), or from a specific starting point to a target. The execution path followed during the trace function is automatically saved into a MARK path called TRACK. This path has a single entry and exit beginning with the line where a trace function starts and ending where the trace function is discontinued. TRACK is valid until a new trace function is initiated.

Unconditional PERFORMs can be included in TRACK without following a path into the PERFORMed unit by using the BYPASS PERFORMED CODE decision option. This method shortens the trace function without a detailed trace of the PERFORMed unit.

Using the trace function is a convenient way to view paths in logical execution order. All executable statements on the path from the starting point to where the trace function stops are highlighted. A long message indicates the reason for stopping. For a decision point, the line number where the trace function stopped and the line number to continue with show.

Columns 73 through 80 contains these information tags on a traced line:

DECISION	for a decision point
STOP PNT	for an intermediate stopping point
START	for the beginning of the trace function
OPTION $n$	for an option choice
TARGET	for a destination of the trace function

Multiple information tags appearing on a line are separated with a slash (/).

The trace function stops and redisplay the Source View screen when these occur:

- A target is reached.
- A decision point is reached. All valid options are presented in columns 73 through 80 and line three of the Source View screen.
- Before entering or exiting a PERFORMed unit (DECISION).
- At a GOTO statement that transfers to code not displayed on the screen (STOP PNT).
- At a GOTO or EXIT from PERFORMs in the forward direction, or a LABEL in the backward direction (STOP PNT).

- The bottom of the screen is reached in the NEXT or FORWARD direction, or the top of the screen is reached in the PREVIOUS or BACKWARD direction (STOP PNT).
- An apparent decision point is reached but only one choice exists. This is shown as a stopping point rather than a decision point.

To continue the trace function, type the desired option in the command input area and press Enter.

**Note:** \_\_\_\_\_

See [Chapter 5, "Logic," on page 99](#) or the online help for more information about TRACKs and the Logic Order Search pop-ups.

---

## *Examples*

**Example 1.** This command traces forward to the first EXIT in the program:

```
TRACE EXITS
```

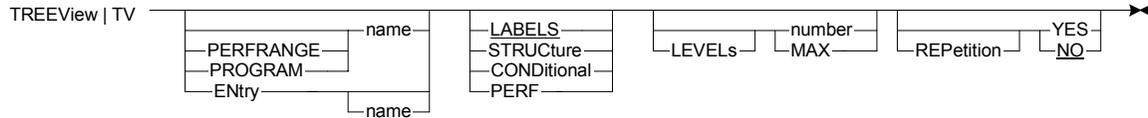
To highlight the resulting MARK path, type this command:

```
FX TRACK
```

**Example 2.** This command traces forward through SUBNET1, that is a part of the NETWORK created by the FLOW command:

```
TRACE SUBNET1
```

## TREEVIEW Command



### Function

The TREEVIEW command displays the Tree View screen, that shows a structural program representation in logical execution order. This command can also be executed on the Tree View screen to end the current Tree View and start a new Tree View with the specified targets.

### Operands

Operand	Description
Blank	Specifies that the defaults are used.
PERFRANGE <i>name</i>	Specifies the starting paragraph for the Tree View display. A name is any PROCEDURE DIVISION paragraph name or section name. The PROCEDURE literal can also be specified.
PROGRAM <i>name</i>	Specifies the main program or any of the nested programs representing all the code that is executed by entering the specified program.  For COBOL II Release 3 or later programs, a data item, label, or program name that may be ambiguous or used multiple times can be qualified by using OF followed by the program name. For example, if the label P120-READ exists in another program and also in the program VIAIDEM1, then it can be qualified as P120-READ OF VIAIDEM1.
ENTRY <i>name</i>	Displays the main program structure and the structure for all ENTRY points. If a name is specified, the structure for that ENTRY point displays.
LABELS	Generates a Tree View display of the structure of the program using the paragraphs and section labels of the control flow statements. LABELS is the default.
STRUCture	Generates a Tree View display of the structure of the program using control flow verbs (PERFORM, GO TO, ALTER, etc.) and program labels.

Operand	Description
CONDitional	Generates a Tree View display of the conditional statements that surround control flow statements, plus the control flow statements and labels.
PERF	Generates a Tree View display of the perform range hierarchy for the program.
LEVELs <i>number</i>	Controls the amount of information displayed on entering Tree View. Specify the number of nested control flow structures to be displayed. The default, 2, displays the two levels of nested performs.
MAX	Displays the maximum nesting level. However, if the maximum nesting level consists of only repeated PERFORM ranges, then the nested level is truncated to the next higher level that does not consist exclusively of repeated PERFORM ranges.
REPetition	Controls the display of duplicate perform ranges. If YES is specified, all duplicate perform ranges display. If NO is specified, duplicate perform ranges are only flagged as REPEATED and are not expanded unless you type ZOOMIN for the perform ranges. The default is NO.

### Usage Notes

The Tree View screen can also be displayed by selecting View ► Tree and pressing Enter.

## VIEW Command

View 

### Function

The VIEW command is used to start a new Source View session for the specified program while retaining the current session.

### Operands

Operand	Description
Blank	Displays the AKR Program Selection pop-up for selection of a program.
<i>pgm</i>	Specifies a program a Source View session is to be started for.

### Usage Notes

When finished with the requested program, the END or GOBACK command returns to the original session.

## ZOOMIN/ZOOMOUT Commands

ZOOMIn | ZI  

ZOOMOut | ZO  

### Function

The ZOOMIN and ZOOMOUT commands are used to display or exclude source lines according to the hierarchical levels of the program.

### Operands

None. These commands are entered in the command input area with no operands and are dependent on the cursor location.

### Usage Notes

ZOOMIN and ZOOMOUT are also available on the View pull-down. See the online help for more information.

The ZOOMIN and ZOOMOUT commands show the structure of a program and provide a means of stepping through each level or going directly into or out of a particular section of source code. The ZI and ZO line commands can also be used to perform the ZOOMIN and ZOOMOUT functions.

Program level hierarchy consists of the IDENTIFICATION DIVISION statement, the PROGRAM-ID statement, and the END PROGRAM statement, if any. This is especially useful in revealing the hierarchy of nested programs that is possible with COBOLII Release 3.

PROCEDURE DIVISION hierarchy consists of section labels, paragraph labels, and paragraph code. DATA DIVISION hierarchy consists of sections, FDs, 01, or 77 levels, and all definitions within an 01 level.

For example, begin by excluding all lines from the screen, then type ZOOMIN. The program level information is shown including the program name and its last physical line. For COBOLII Release 3 modules with sub-programs, this information shows for each of the programs. Type ZOOMIN again. The PROGRAM-ID and END-PROG display first then the DIVISION headings are shown. Type ZOOMIN in the command input area and place the cursor on the PROCEDURE DIVISION statement, then press Enter. (Or, place the cursor on the PROCEDURE DIVISION statement, then press PF07/19.) All PROCEDURE DIVISION headings display.

To see the source within a particular heading, scroll to it and type `ZOOMIN` again. The paragraph names display. Continue following this process to see the code and copy statements within several paragraphs. Type `ZOOMOUT` on a paragraph name to exclude it from the screen after it is viewed. All source lines under the paragraph name are excluded from the screen.

These are the general guidelines for `ZOOMIN`:

- If the cursor is on an excluded block of lines, `ZOOMIN` displays the highest structure level within that block of lines. If there are multiple statements at that level, they also display.
- If `ZOOMIN` is issued repeatedly from the same cursor position, successive heading levels display.

These are the general guidelines for `ZOOMOUT`:

- The cursor cannot be positioned on an excluded line when `ZOOMOUT` is entered.
- `ZOOMOUT` excludes the levels in a lowest to highest order. If `ZOOMOUT` is issued repeatedly from the same cursor location, successive heading levels are excluded.

## Line Commands

Line commands are entered over the line numbers in the prefix area on the screen. Line commands are either single format or block format. Single format refers to a line command being entered on an individual line. [Figure 104](#) illustrates the line command to exclude line 332.

**Figure 104 • Single Format Line Command Example**

```
000331      MOVE ZIP-CODE TO HLD-ZIP.
X00332      IF HLD-ZIP-PREFIX EQUAL CUR-PREFIX
000333      NEXT SENTENCE
```

A number can be included with some single format commands to indicate a specified number of lines to be processed using the same command. Processing begins with the line where the command is entered, and includes the current line and subsequent lines as indicated by the number specified. If the specified number of lines exceeds the available source lines, all remaining lines are processed. The maximum you can enter is 99999.

[Figure 105](#) illustrates the single format Exclude line command followed by a number to exclude lines 332 through 336.

**Figure 105 • Single Format Line Command with a Number Specified**

```
000331      MOVE ZIP-CODE TO HLD-ZIP.  
X50332      IF HLD-ZIP-PREFIX EQUAL CUR-PREFIX  
000333      NEXT SENTENCE  
000334      ELSE  
000335      PERFORM P150-SUBTOT  
000336      THRU P169-EXIT  
000337      MOVE HLD-ZIP-PREFIX TO CUR-PREFIX
```

Block format refers to double character line commands entered on multiple lines. Block line commands process for all lines between (and including) the lines containing the commands. [Figure 106](#) illustrates the block format Exclude Block command being used to exclude lines 332-336.

**Figure 106 • Block Format Line Command Example**

```
000331      MOVE ZIP-CODE TO HLD-ZIP.  
XX0332      IF HLD-ZIP-PREFIX EQUAL CUR-PREFIX  
000333      NEXT SENTENCE  
000334      ELSE  
000335      PERFORM P150-SUBTOT  
XX0336      THRU P169-EXIT  
000337      MOVE HLD-ZIP-PREFIX TO CUR-PREFIX
```

Line commands can be removed by typing over them with spaces, pressing ERASE EOF with the cursor at the beginning of the command, or by typing RESET in the command input area.

Line commands can be ambiguous when entered. [Figure 107](#) illustrates a common ambiguity.

**Figure 107 • Ambiguous Line Command Example**

```
024500      MOVE ZIP-CODE TO HLD-ZIP.  
X24600      IF HLD-ZIP-PREFIX EQUAL CUR-PREFIX  
024700      NEXT SENTENCE
```

It is unclear whether two lines, 24 lines or 246 lines are to be excluded. If two lines are to be excluded, one or more spaces should be entered following the X2. The ERASE EOF key could be used to erase the numbers following the X2.

## F (First) Line Command

Fn

### Function

The F (First) line command redisplay the specified number of excluded lines. These lines redisplay starting with the first line in the block of excluded lines.

### Operands

**n.** The number of excluded lines to redisplay. The default is 1. If the number specified is greater than the number of excluded lines, all lines in the excluded block redisplay.

### Usage Notes

The F (First) line command redisplay lines excluded as the result of any primary or line command that excludes lines from the screen display.

### Example

[Figure 108](#) and [Figure 109](#) illustrate the first two lines in the excluded block being redisplayed.

**Figure 108 • F (First) Line Command Example**

```
000002  PROGRAM-ID.  PRDEMO.
F2      - - - - - - - - - - - - - 5 LINES NOT DISPLAYED
000008  INPUT-OUTPUT SECTION.
```

**Figure 109 • F (First) Line Command Results**

```
000002  PROGRAM-ID.  PRDEMO.
000003  AUTHOR.  WRITTEN BY ASG AT LANGLVL 2.
000004  ENVIRONMENT DIVISION.
- - - - - - - - - - - - - 3 LINES NOT DISPLAYED
000008  INPUT-OUTPUT SECTION.
```

## H (Highlight) Line Command

Hn

### Function

The H (Highlight) line command is used to highlight a line or group of lines.

### Operands

**n.** The number of consecutive lines to highlight. The default is 1. If the number specified is greater than the number of available lines, all remaining lines are highlighted.

### Usage Notes

The H (Highlight) line command is used to highlight a contiguous group of lines.

### Example

[Figure 110](#) and [Figure 111](#) illustrate the Highlight line command being used to highlight three source code lines.

**Figure 110 • H (Highlight) Line Command Examples**

```
H3 281   CALL 'VIAIDEM1' USING MASTER-IN
000282                                     MASTER-END-OF-FILE
000283                                     MASTER-REPORT-DATE.
000284
000285   PERFORM P100-PRINT
000286           THRU P119-EXIT.
```

**Figure 111 • H (Highlight) Line Command Results**

```
000281   CALL 'VIAIDEM1' USING MASTER-IN
000282                                     MASTER-END-OF-FILE
000283                                     MASTER-REPORT-DATE.
000284
000285   PERFORM P100-PRINT
000286           THRU P119-EXIT.
```

## HH (Highlight Block) Line Command

HH

### Function

The HH (Highlight Block) line command is used to highlight a block of lines.

### Operands

None.

### Usage Notes

The HH (Highlight block) line command is entered in the prefix area on the first and last line of the block to be highlighted.

### Example

[Figure 112](#) and [Figure 113](#) illustrate the HH line command being used to highlight three source code lines.

**Figure 112 • HH (Highlight Block) Line Command Example**

```
HH0281  CALL 'VIAIDEM1' USING MASTER-IN
000282                                MASTER-END-OF-FILE
HH0283                                MASTER-REPORT-DATE.
000284
000285  PERFORM P100-PRINT
000286                                THRU P119-EXIT.
```

**Figure 113 • HH (Highlight Block) Line Command Results**

```
000281  CALL 'VIAIDEM1' USING MASTER-IN
000282                                MASTER-END-OF-FILE
000283                                MASTER-REPORT-DATE.
000284
000285  PERFORM P100-PRINT
000286                                THRU P119-EXIT.
```

## J (Jump) Line Command

J

### Function

The J (Jump) line command is used to return to the Source View screen at the location corresponding to the line where this command is issued on the Tree View screen.

### Operands

None.

### Usage Notes

Enter the J (Jump) line command on the line that you want to return to.

## Label Line Command

.label

### Function

The Label line command is used to assign a name to a particular source line.

### Operands

**.label.** A period followed by 1 to 5 alphabetic characters to be used as the name for the line where it is entered.

### Usage Notes

You can use the LOCATE primary command followed by a label name to scroll to the labelled line.

### Example

[Figure 114](#) illustrates the .label line command being used to label line 281 to EQL. Line 281 can be referenced with EQL as well as line 281.

**Figure 114 • Label Line Command Example**

```
.EQL81      CALL 'VIAIDEM1' USING MASTER-IN
000282                MASTER-END-OF-FILE
000283                MASTER-REPORT-DATE.
000284
000285      PERFORM P100-PRINT
000286                THRU P119-EXIT.
000287
```

## L (Last) Line Command

*L**n*

### Function

The L (Last) line command redisplay the specified number of excluded lines. These lines redisplay starting with the last line in the block of excluded lines.

### Operands

*n*. The number of excluded lines to redisplay. The default is 1. If the number specified is greater than the number of excluded lines, all lines in the excluded block redisplay.

### Usage Notes

The L (Last) line command redisplay lines excluded as the result of any primary or line command that excludes lines from the screen display.

### Example

[Figure 115](#) and [Figure 116](#) illustrate the L line command being used to redisplay the last two lines in the excluded block.

**Figure 115 • L (Last) Line Command Example**

```
000002  PROGRAM-ID.  VIAIDEMO.
L2      - - - - - - - - - - - - - - - - - - - - 5 LINES NOT DISPLAYED
000008  INPUT-OUTPUT SECTION.
```

**Figure 116 • L (Last) Line Command Results**

```
000002  PROGRAM-ID.  VIAIDEMO.
- - - - - - - - - - - - - - - - - - - - 3 LINES NOT DISPLAYED
000006  SOURCE-COMPUTER.  IBM-370.
000007  OBJECT-COMPUTER.  IBM-370.
000008  INPUT-OUTPUT SECTION.
```

## S (Show) Line Command

*Sn*

### Function

The S (Show) line command is used to redisplay the specified number of excluded lines. These lines redisplay starting with the first line in the excluded block.

### Operands

**n.** The number of consecutive excluded lines to show. The default is 1. If the number specified is greater than the number of excluded lines, all lines in the excluded block redisplay.

### Usage Notes

The S (Show) line command redisplay lines excluded as the result of any primary or line command that excludes lines from the screen display.

### Example

[Figure 117](#) and [Figure 118](#) illustrate the S line command being used to redisplay two excluded source lines.

**Figure 117 • S (Show) Line Command Example**

```
000002 PROGRAM-ID. VIAIDEMO.
S2      - - - - - 5 LINES NOT DISPLAYED
000008 INPUT-OUTPUT SECTION.
```

**Figure 118 • S (Show) Line Command Results**

```
000002 PROGRAM-ID. VIAIDEMO.
000003 AUTHOR. WRITTEN BY ASG AT LANGLVL 2.
000004 ENVIRONMENT DIVISION.
- - - - - 3 LINES NOT DISPLAYED
000008 INPUT-OUTPUT SECTION.
```

## SS (Show Block) Line Command

SS

### Function

The SS (Show block) line command is used to redisplay a block of excluded lines.

### Operands

None.

### Usage Notes

The SS (Show block) line command is entered in the line command area on the first and last line of the excluded blocks to be redisplayed.

### Example

[Figure 119](#) and [Figure 120](#) illustrate the SS line command being used to redisplay two blocks of excluded lines.

**Figure 119 • SS (Show block) Line Command Example**

```
000401 C4H-TO-NUM.
SS0402 * MOVE COMP-4 (HALFWD) TO NUMERIC (UNSIGNED)
- - - - - 4 LINES NOT DISPLAYED
000407 * MOVE COMP-4 (HALFWD) TO NUMERIC (SIGNED)
- - - - - 4 LINES NOT DISPLAYED
SS0412 C4H-TO-C1.
```

**Figure 120 • SS (Show Block) Line Command Results**

```
000401 C4H-TO-NUM.
000402 * MOVE COMP-4 (HALFWD) TO NUMERIC (UNSIGNED)
000403     MOVE N-C4HALF   TO 77-NU.
000404     MOVE N-C4HALFS  TO 77-NU.
000405     MOVE X-C4HALF   TO 77-NU.
000406     MOVE X-C4HALFS  TO 77-NU.
000407 * MOVE COMP-4 (HALFWD) TO NUMERIC (SIGNED)
000408     MOVE N-C4HALF   TO 77-NUS.
000409     MOVE N-C4HALFS  TO 77-NUS.
000410     MOVE X-C4HALF   TO 77-NUS.
000411     MOVE X-C4HALFS  TO 77-NUS.
000412 C4H-TO-C1.
```

## X (Exclude) Line Command Function

Xn

The X (Exclude) line command is used to exclude a line or group of lines from being displayed. Excluded lines display as a row of dashes with the number of excluded lines indicated.

### Operands

**n.** The number of consecutive lines to exclude. The default is 1. If the number specified is greater than the number of available lines, all remaining lines are excluded.

### Usage Notes

Excluded lines display as a row of dashes with the number of excluded lines indicated.

### Example

[Figure 121](#) and [Figure 122](#) illustrate the X line command being used to exclude lines 332-336.

**Figure 121 • X (Exclude) Line Command Example**

```
000331      MOVE ZIP-CODE TO HLD-ZIP.  
X5 332      IF HLD-ZIP-PREFIX EQUAL CUR-PREFIX  
000333          NEXT SENTENCE  
000334      ELSE  
000335          PERFORM P150-SUBTOT  
000336              THRU P169-EXIT  
000337      MOVE HLD-ZIP-PREFIX TO CUR-PREFIX
```

**Figure 122 • X (Exclude) Line Command Results**

```
000331      MOVE ZIP-CODE TO HLD-ZIP.  
- - - - - 5 LINES NOT DISPLAYED  
000337      MOVE HLD-ZIP-PREFIX TO CUR-PREFIX
```

## XX (Exclude Block) Line Command

XX

### Function

The XX (Exclude Block) line command is used to exclude a block of lines from being displayed. Excluded lines display as a row of dashes with the number of excluded lines indicated.

### Operands

None.

### Usage Notes

The XX (Exclude block) line command is entered in the line command area on the first and last line of the block of lines to be excluded from being displayed.

### Example

[Figure 123](#) and [Figure 124](#) illustrate the XX line command being used to exclude a block of five lines.

**Figure 123 • XX (Exclude Block) Line Command Example**

```

000331      MOVE ZIP-CODE TO HLD-ZIP.
XX0332      IF HLD-ZIP-PREFIX EQUAL CUR-PREFIX
000333              NEXT SENTENCE
000334      ELSE
000335              PERFORM P150-SUBTOT
XX0336              THRU P169-EXIT
000337      MOVE HLD-ZIP-PREFIX TO CUR-PREFIX

```

**Figure 124 • XX (Exclude Block) Line Command Results**

```

000331      MOVE ZIP-CODE TO HLD-ZIP.
- - - - - 5 LINES NOT DISPLAYED
000337      MOVE HLD-ZIP-PREFIX TO CUR-PREFIX

```

## ZI (Zoom In) Line Command

ZI

Tree View only

### Function

The ZI (Zoom in) line command is used to redisplay excluded source lines according to the hierarchical levels in the program. The ZI line command is used in conjunction with the ZO (Zoom out) line command to show the structure of a program and provide a means of stepping through each level or going directly into or out of a particular section of source code.

### Operands

None.

### Usage Notes

The ZI (Zoom in) line command is used to show the structure of a program. PROCEDURE DIVISION hierarchy consists of section labels, paragraph labels, and paragraph code. DATA DIVISION hierarchy consists of sections, FDs, 01, or 77 levels, and all definitions within an 01 level.

If the ZI (Zoom in) line command is entered on an excluded block of lines, the highest structure level within that block of lines displays. If there are multiple statements at that level, they also display.

**Note:** \_\_\_\_\_

See "[ZOOMIN/ZOOMOUT Commands](#)" on page 324 for more information about displaying and excluding source lines according to the hierarchical levels of the program.

---

## ZO (Zoom Out) Line Command

ZO

Tree View only

### Function

The ZO (Zoom out) line command is used to redisplay a source line in its original format after a Zoom line command is entered. The ZO (Zoom out) line command is used in conjunction with the ZI (Zoom in) line command to show the structure of a program and provide a means of stepping through each level or going directly into or out of a particular section of source code. The ZI (Zoom in) and ZO (Zoom out) line commands are used to display or exclude source lines according to the hierarchical levels of the program.

### Operands

None.

### Usage Notes

The ZO (Zoom out) line command can only be entered on a line that displayed using the ZI (Zoom in) line command. See ["ZOOMIN/ZOOMOUT Commands" on page 324](#) for more information about displaying and excluding source lines according to the hierarchical levels of the program.

The ZO (Zoom out) line command can also be used to remove lines that were displayed in a window on the screen, as the result of a ZI (Zoom in) line command.



---

# 11

## Analyze

---

This chapter describes the analyze process and contains these sections:

Topic	Page
<a href="#">Overview</a>	<a href="#">339</a>
<a href="#">Analyzing a COBOL Program</a>	<a href="#">340</a>
<a href="#">The Analyze Process</a>	<a href="#">342</a>
<a href="#">Automatic JCL Modifications</a>	<a href="#">349</a>
<a href="#">Analyze Summary Report</a>	<a href="#">352</a>
<a href="#">Analyze Options</a>	<a href="#">354</a>

### Overview

A program must be analyzed before the ESW products can provide any information about it. This chapter describes the methods used to analyze programs.

The analyze process gathers information about the program, including program relationships, logic, data, and execution paths, and stores this information in the AKR. After the analyze information is placed in the AKR, it is available to ESW products in online and batch environments, where it is accessed to provide valuable information about the design and operation of user systems.

## Analyzing a COBOL Program

The analyze process is similar to a COBOL compile. The process has these three primary inputs:

- Source COBOL program (including copy books)
- JCL used to compile and link the COBOL program
- Options and features that tailor the analyze steps

### Analyze Input Descriptions

#### COBOL Source Program

The analyze process requires these basic program standards:

- The COBOL language as specified in the *IBM OS/VS COBOL, and COBOL/370 Language Reference Manuals* is accepted by the analyze job. It correctly processes any program that can be compiled without warnings or errors by the IBM OS/VS COBOL or COBOL II compilers.
- OS/VS COBOL programs that receive conditional (C), error (E), or disaster (D) messages from the IBM compiler cannot be successfully analyzed.
- COBOL/370 programs that receive error (E), severe (S), or unrecoverable (U) messages from the IBM compiler cannot be successfully analyzed.

This table contains the Program Analyzer resource estimates to process COBOL programs of various sizes:

Insight Analyze Resource Estimates					
Source Lines	Virtual Memory Size	XA Memory Size	CPU Time MM:SS	AKR Blocks	VIAUT2 Cyls
1000	1060K	12100K	0:04	150	2
2000	1060K	12200K	0:10	300	3
5000	1060K	12500K	0:30	750	6
10000	1060K	12800K	1:00	1500	12
20000	1060K	14000K	2:00	3000	24
50000	1060K	16000K	5:00	6000	50

This information is the result of running Analyze under these criteria:

Version:	Center R4.0
CPU Type:	3090-600 running MVS/ESA
Disk Type:	3390
Analyze Params:	BUFMAXK=4096K
Compiler Params:	BUF=256K,SIZ=1024K

### *Compile/Link JCL*

This JCL should be the complete JCL used to compile the program. Specifically, the JCL should contain steps to fetch the source from the source manager (such as Librarian or Panvalet), execute the preprocessor(s), invoke the compiler with the appropriate options and COPY libraries, then, if applicable, invoke the linkage editor.

### *Analyze Features and Options*

The analyze features indicate the type of analysis to be performed. An Encore analysis provides the information required for code isolation and execution flow capabilities. A SmartTest analysis provides the testing and debugging information required by SmartTest. An Extended SmartTest analysis provides comprehensive program analyzing capabilities in addition to the testing and debugging capabilities of a SmartTest analysis. An Insight analysis provides logic and execution flow capabilities. A SmartDoc analysis provides the information required for SmartDoc reports. A SmartDoc Extended analysis provides data flow analysis.

Default options for the analyze process are established at installation time. Options that are to be overridden are specified when submitting the analyze job.

## The Analyze Process

The analyze process consists of setting up and executing a batch job. There are three methods used to invoke the analyze process. The method you used depends primarily on the environment from where the analyze process is invoked, but may depend on the access method containing the compile/link JCL. These are the three methods used to invoke the analyze process:

Method	Description
File - Analyze Submit Pop-up	Select File ► Analyze action from the File pull-down, or type ANALYZE in any command input area, to display the Analyze Submit pop-up. Type the required input and output information and submit the job. See the online help for information about the File - Analyze Submit pop-up.
ISPF	From any ISPF screen, execute the VIASUBDS CLIST. To execute this CLIST, type this command:  <code>TSO VIASUBDS <i>dsn</i> <i>parms</i></code> where:  <i>dsn</i> is a PDS member or sequential dataset containing the compile JCL.  <i>parms</i> represents any of the available execution parameters described in the VIASUBDS and VIASUB parameter tables.
ISPF/PDF Edit	Execute the VIASUB PDS edit macro. To execute this edit macro type VIASUB <i>parms</i> , where <i>parms</i> represents any of the available execution parameters described the VIASUBDS and VIASUB parameter tables.

This table lists the methods for executing an analyze job and when to use each.

Compile JCL is From	Method for Executing Analyze Job
PDS or sequential dataset	Analyze Submit pop-up, VIASUBDS CLIST, or VIASUB edit macro
Librarian, Panvalet, or other user source manager when editing the JCL with ISPF/PDF	VIASUB edit macro
Screen-driven submit facility that generates JCL	VIASUBDS CLIST

## Analyze Using ISPF

You can use the VIASUBDS CLIST to submit the analyze job from any ISPF screen. This is the syntax for VIASUBDS:

```
TSO VIASUBDS input.jcl.dsn parms
```

where:

*input.jcl.dsn* is the dataset containing the compile/link JCL. This dataset must be a sequential dataset or a member of a PDS.

*parms* is one or more parameters that control the operation of VIASUBDS. Typically, you enter the PANEL parameter to display the Analyze Submit Parameters screen in order to enter any necessary parameters. The parameters are saved in the ISPF profile and used as defaults for the next analyze submission. The VIASUBDS and VIASUB Parameters table contains a list of these parameters, with default parameters underlined.

**Note:** \_\_\_\_\_

Using the VIASUBDS CLIST requires the ESW CLIST library to be available through the standard SYSPROC allocations.

\_\_\_\_\_

## Analyze Using ISPF/PDF Edit

You can use the VIASUB edit macro to submit the analyze job from the ISPF/PDF Edit screen. This is the syntax for VIASUB:

```
VIASUB parms
```

where *parms* is one or more parameters that control the operation of VIASUB. Typically, you enter the PANEL parameter to display the Analyze Submit Parameters screen in order to enter any necessary parameters. The parameters are saved in the ISPF profile and used as defaults for the next analyze submission. The VIASUBDS and VIASUB Parameters table contains a list of these parameters, with default parameters underlined.

**Note:**

Using the VIASUB edit macro requires the ESW CLIST library to be available through the standard SYSPROC allocations.

---

Parameter	Description
AKR(#####)	Indicates the AKR where the analyze job results are placed. The specified name must conform to the standard TSO dataset naming conventions.
AOPT(#####)	Specifies the options to be supplied to the analyze job. The COBOLII option is automatically added if the compiler you specified in the input JCL is COBOL II. When specifying more than one analyze option, the options should be separated by commas and enclosed in single quotes; for example, AOPT('XMEM,RECUR,SUBSYS=D239'). See " <a href="#">Analyze Options</a> " on page 354 for information about each analyze option.
CMPL NOCMPL	CMPL indicates that a COBOL compile and an analysis is to be executed by the new JCL. NOCMPL indicates that the new JCL is to bypass the compile step and only execute the analyze job. When you specify NOCMPL, a return code of 1000 (decimal) greater than the analyze return code is produced. This causes the subsequent job steps (e.g., a link edit) to be bypassed based on a successful compilation. You cannot specify NOCMPL if a SmartTest analysis is being executed.
DSCHK NODSCHK	DSCHK indicates that the datasets needed by the resulting JCL are to be verified to ensure they exist. Specifically, the AKR and the load library containing VIASMNTR are checked. When you specify NODSCHK, the AKR and the load library need not exist at the time VIASUB or VIASUBDS is executed. NODSCHK is useful when you are preparing the JCL for submission on another system, or for delaying execution when an AKR does not yet exist. Notice that the cataloged procedure libraries must exist and be accessible to VIASUBDS or VIASUB.
EDIT	EDIT specifies that the resulting JCL is not to be submitted for batch processing. The PDF editor is invoked for the resulting JCL. You can make any desired changes and submit the JCL by typing SUBMIT. EDIT must be entered each time it is needed. Notice that the edits made to the JCL are not saved. To save the modified JCL elsewhere, you must use the CREATE command. The EDIT option is ignored if the Analyze Submit Parameters screen displays. In this case, type E to edit the JCL.
INS NOINS	INS specifies that an Insight analysis is to be performed.

---

Parameter	Description
OUTPUT(#####)	Specifies that the resulting JCL is not to be submitted for batch processing. The JCL is written to the specified dataset. The specified name must conform to the standard TSO dataset naming conventions. A dataset is created if it does not already exist. OUTPUT must be entered each time it is needed.
PANEL NOPANEL	PANEL indicates that the Analyze Submit Parameters screen is to be displayed allowing you to enter parameters for the analyze job. The Analyze Submit Parameters screen displays even if you specify a valid AKR name as a parameter, or you can access it from the ISPF profile when you specify PANEL.
PGM(#####)	Specifies a name to be used when storing the program in the AKR. This name overrides the program name in the PROGRAM-ID paragraph.
PROONLY	Indicates the JCL contains only a cataloged procedure rather than a complete job. PROONLY suppresses the generation of the VIAIN DD statement. You must enter PROONLY each time it is needed.
REUS NOREUS	REUS specifies that when the program is tested using SmartTest, it is dynamically loaded and tested with RUN NOMONITOR.
ENS NOENS	ENS specifies that an Encore analysis is to be performed.
SD NOSD	SD specifies that a SmartDoc analysis is to be performed.
SDR NOSDR	SDR specifies that SmartDoc reports are run.
SDX NOSDX	SDX specifies that a SmartDoc Extended analysis is to be performed.
ST NOST	ST specifies that a SmartTest analysis is to be performed.
STX NOSTX	STX specifies that an Extended SmartTest analysis is to be performed. When you specify the INS and ST parameters, an Extended SmartTest analysis is automatically performed.

## Analyze Submit Parameters Screen

The Analyze Submit Parameters screen ([Figure 125](#)) displays when you specify the PANEL parameter when executing VIASUBDS or VIASUB, or when you use the NOPANEL option and an error condition is detected.

**Figure 125 • Analyze Submit Parameters Screen**

```

ASG-ESW - Prepare Program
Command ==> _____
                E - Edit JCL      S - Submit JCL      D - Doc Options

Compile and link JCL (PDS or sequential):
  Data set name 'USER.TEST.CNTL(YOURJCL)'

Analyze features (Y/N):
  Understand: Y   Test: Y   Extended Analysis: Y   Document: N
  Re-engineer: N
  AKR data set name 'USER.TEST.AKR'
  AKR program name _____ (if overriding PROGRAM-ID)

Analyze options:
  _____
  _____

Compile? (Y/N) . . . . . Y      (Y if needed by features)
Link load module reusable? (Y/N) Y      (Test only)
    
```

## Options

Options	Description
E - Edit JCL	<p>Enables you to review or change the compile/analyze JCL, if necessary. When you select the E option, the JCL to be edited is generated from the JCL you specified when the VIASUBDS CLIST or VIASUB edit macro was invoked, applying the rules outlined in the Automatic JCL Modifications section. The generated JCL is then displayed on the Edit screen.</p> <p>When you finish editing, type ISPF SUBMIT to submit the edited JCL for execution. Optionally, you can save the edited JCL in a partitioned dataset by using the ISPF CREATE command. Otherwise, any changes made at this time are not saved.</p>
S - Submit JCL	<p>Submits the JCL to compile/analyze the specified program. The JCL submitted is generated from the JCL you specified when the VIASUBDS CLIST or VIASUB edit macro was invoked, applying the rules outlined in the Automatic JCL Modifications section.</p>
D - Doc Options	<p>Displays only if SmartDoc is installed. Type D to display the SmartDoc Options screen that is used to request an Extended SmartDoc analysis and to specify what reports (if any) to generate.</p>

## Fields

Field	Description	
Analyze feature	Understand	Displays only if Insight is installed. This type of analysis provides the logic and program execution flow capabilities of Insight. If Insight is the only product installed, this field contains YES and cannot be changed. The default is Y.
	Test	Displays only if SmartTest is installed. Y indicates that a SmartTest compile/analysis is to be performed. This type of analysis provides the testing and debugging information required by SmartTest. If SmartTest is the only product installed, this field contains YES and cannot be changed. The default is Y.
	Extended Analysis	Displays only if SmartTest is installed. This type of analysis provides comprehensive program analyzing capabilities in addition to the testing and debugging capabilities of SmartTest. The default is Y.  A SmartDoc Extended analysis is specified on the SmartDoc Options screen.
	Document	Displays only if SmartDoc is installed. This type of analysis provides the report information generated by SmartDoc. If SmartDoc is the only product installed, this field contains YES and cannot be changed. The default is N.
	Reengineer	Displays only if Encore is installed. ES specifies that an Encore compile/analysis is to be performed. This type of analysis provides the logic and program execution flow capabilities of Encore. If Encore is the only product installed, this field contains YES and cannot be changed. The default is N.
AKR data set name	Specifies the name of the AKR containing the analyzed program information.	

Field	Description
AKR program name	<p>Specifies the alias name to be used by the analyze process to save its results in the AKR.</p> <p>If you do not enter a value in this field, the analyze job saves the results in the AKR using the program name from the PROGRAM-ID statement in the COBOL source.</p> <p>If you enter an AKR program name, the analyzed program is saved in the AKR as the entered name, and also as an alias of the PROGRAM-ID.</p> <p>If a program contains ENTRY points, the analyze job is also saved in the AKR a member for each ENTRY point, with an alias of the PROGRAM-ID.</p> <p><b>Note:</b></p> <p>This field is only used for the AKR program name and does not change the COBOL program name in the source.</p>
Analyze options	<p>Specifies the analyze options that are to be overridden. Default options for the analyze job are established at installation time. Analyze options that can be entered in this field are described in the Analyze Options appendix.</p>
Compile?	<p>A program does not need to be compiled if Insight, Encore, or SmartDoc are the only features specified. The compile step can be suppressed by typing N in this field. If you select SmartTest and/or Extended analysis, this field is forced to a value of Y.</p>
Link load module reusable?	<p>Tests a program under SmartTest that is dynamically loaded if it is tested. It is necessary to mark the load module as reusable so that the Breakpoints are retained across calls. The default is Y</p>
Display this panel by default in the future?	<p>Causes the ISPF profile to be updated to display this screen whenever subsequent executions of VIASUBDS or VIASUB are invoked. If you specify N, this screen is not displayed on subsequent executions of VIASUBDS or of VIASUB unless an error condition is encountered.</p>

## Automatic JCL Modifications

The analysis process automatically modifies the JCL based on the specified parameters and analyze options. If problems arise, you can use this procedure as a checklist to perform the analyze process manually until you determine and resolve the problem. You can make changes to the JCL, the compile procedure, or to a copy of the compile procedure.

### *To manually modify the JCL*

- 1 Replace these PGM= parameters in the compile step(s):

PGM= parameter	Replace with...
PGM=IKFCBL00	PGM=VIACOBVS
PGM=IGYCRCTL	PGM=VIACOBII
PGM=CPXUPTSM	PGM=VIAOPT3
PGM=CAOTSMON	PGM=VIAOPTII

- 2 Add DD statements to the compile step(s) for these datasets:

```
//VIADCTOC DD SYSOUT=*
//VIADCRPT DD SYSOUT=*
//VIALOG DD SYSOUT=*
//VIAMRPT DD SYSOUT=*
//VIAPRINT DD SYSOUT=*
//VIAAKR DD DSN=[specified AKR name],DISP=SHR
```

- 3 If the SYSIN DD statement contains FREE=CLOSE, change it to FREE=END.
- 4 Ensure that the ESW load libraries are available to the modified step by adding a //STEPLIB DD statement specifying the ESW load libraries, or by concatenating these libraries to an existing STEPLIB DD.
- 5 Ensure that the JOB and the modified STEP EXEC statements have a minimum of REGION=4096K.
- 6 Add a VIAIN DD statement that designates the features and options to be used during analysis. This is the format:

```
//VIAIN DD *
* ANALYZE FEATURES:
INS,ST,STX,SD,SDX,SDR,RNS
/*
```

You can also preform this step manually by modifying the COBOL parameter string to include the appropriate ESW parameter, for example:

```
VPARAM=(vopt, vopt, vopt...)
```

where *vopt* can be:

<i>vopt</i>	Indicating...
INS	An Insight only analysis (no COBOL compile)
ST	A SmartTest only analysis (no Extended analysis)
STX	A SmartTest Extended analysis
SD	A SmartDoc analysis
SDX	A SmartDoc Extended analysis
SDR	A SmartDoc report generation
ENS	An Encore analysis (no COBOL compile)
[analyze parms]	Valid analyze parameters (using the standard analyze options)
CMPL	A COBOL compile (forces a COBOL compile and an analysis to be executed by the JCL)
NOCMPL	Suppress the COBOL compile (JCL bypasses the compile and executes only an analyze job)
(NO)SYSPRINT	Create separate compiler output file
(NO)VIADCOMP	Create SmartDoc intermediate compiler output file. The intermediate compiler output file is used to produce the SmartDoc Compiler Output.
DPARAM	SmartDoc run-time parameters.

**Note:**

If no ESW feature is specified (i.e., INS, ST, STX, SD, SDX, SDR, or ENS), all processing is suppressed. This means the procedure executes a compile as it did before.

---

[Figure 126](#) shows the compile/analyze JCL before submitting the analyze job.

**Figure 126 • Compile/Analyze JCL Before Modifications**

```

// ASG JOB (ASG), 'PANVALET COMPILE'
/*ROUTE PRINT DEST
/* PANVALET EXTRACT
/*
//PANEXT EXEC PGM=PAN#1,REGION=256K
//PANDD1 DD DSN=ASG.COBOL.PANLIB,DISP=SHR
//PANDD2 DD DSN=&&COBIN,UNIT=SYSDA,SPACE=(CYL,(1,1)),
// DISP=(NEW,PASS),DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
++WRITE WORK,VIASDDMO
/*
/*
/* COBOL COMPILE
/*
//COBCOMP EXEC PGM=IKFCBL00,REGION=1024K,COND=(8,LT,PANEXT),
// PARM='SIZE=512K,BUF=128K,LANGLVL(2),LIB,DYNAM'
//STEPLIB DD DSN=SYS1.VSCOBOL.COMPIILER,DISP=SHR
//SYSIN DD DSN=&&COBIN,DISP=(OLD,DELETE)
//SYSLIB DD DSN=ASG.COBOL.COPYLIB,DISP=SHR
//SYSLIN DD DSN=&&LINKIN,UNIT=SYSDA,SPACE=(CYL,(1,1)),
// DISP=(NEW,PASS),DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//SYSPRINT DD SYSOUT=*,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=1573)
//SYSUT1 DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT2 DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT3 DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT4 DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT5 DD UNIT=SYSDA,SPACE=(CYL,(1,1))
/*
/* LINK EDIT
/*
//LINKED EXEC PGM=IEWL,REGION=1024K,COND=(8,LT,COBCOMP)
//SYSLIB DD DSN=SYS1.VSCOBOL.COBLIB,DISP=SHR
//SYSLMOD DD DSN=ASG.BAR.LOAD,DISP=OLD
//SYSPRINT DD SYSOUT=*,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=1573)
//SYSUT1 DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSLIN DD DSN=&LINKIN,DISP=(OLD,DELETE)
// DD *
NAME VIASDDMO(R)
/*

```

[Figure 127 on page 352](#) is an example of the compile JCL illustrated in the previous figure, as it would appear after the compile/analyze JCL is generated according to the rules in this section. Added or modified statements are tagged to the right with ASG NEW and ASG MOD.

Figure 127 • Compile/Analyze JCL After Modifications

```

// ASG JOB (ASG),'PANVALET COMPILE'
//ROUTE PRINT DEST
//*****
//*
//** THIS JCL HAS BEEN MODIFIED BY THE ASG ANALYZE *
//** SUBMIT FACILITY, WHICH CONVERTS COMPILE JCL INTO *
//** COMPILE AND ANALYZE JCL. NEW OR MODIFIED LINES *
//** CONTAIN 'ASG' IN COLUMNS 74 THROUGH 76. *
//**
//*****
//**
//**
//** PANVALET EXTRACT
//**
//PANEXT EXEC PGM=PAN#1,REGION=256K
//PANDD1 DD DSN=ASG.COBOL.PANLIB,DISP=SHR
//PANDD2 DD DSN=&&COBIN,UNIT=SYSDA,SPACE=(CYL,(1,1)),
// DISP=(NEW,PASS),DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
++WRITE WORK,VIASDDMO
/*
//**
//** COBOL COMPILE
//**
//COBCOMP EXEC PGM=VIACOBVS,REGION=4096K,COND=(8,LT,PANEXT),
// PARM='SIZE=512K,BUF=128K,LANGLVL(2),LIB,DYNAM'
//STEPLIB DD DSN=SYS1.VSCOBOL.COMPIER,DISP=SHR
// DD DSN=ASG.VIACENxx.LOADLIB,DISP=SHR
//SYSIN DD DSN=&&COBIN,DISP=(OLD,DELETE)
//SYSLIB DD DSN=ASG.COBOL.COPYLIB,DISP=SHR
//SYSLIN DD DSN=&&LINKIN,UNIT=SYSDA,SPACE=(CYL,(1,1)),
// DISP=(NEW,PASS),DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//SYSPRINT DD SYSOUT=*,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=1573)
//SYSUT1 DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT2 DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT3 DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT4 DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT5 DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//VIADCTOC DD SYSOUT=*
//VIADCRPT DD SYSOUT=*
//VIALOG DD SYSOUT=*
//VIAMRPT DD SYSOUT=*
//VIAPRINT DD SYSOUT=*
//VIAAKR DD DSN=ASG.VIACENxx.AKR,DISP=SHR
//VIAIN DD *
* ANALYZE FEATURES
INS,ST,STX,SD,SDX,SDR,RNS
/*
//**
//** LINK EDIT
//**
//LINKED EXEC PGM=IEWL,REGION=1024K,COND=(8,LT,COBCOMP)
//SYSLIB DD DSN=SYS1.VSCOBOL.COBLIB,DISP=SHR
//SYSMOD DD DSN=ASG.BAR.LOAD,DISP=OLD
//SYSPRINT DD SYSOUT=*,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=1573)
//SYSUT1 DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSLIN DD DSN=&LINKIN,DISP=(OLD,DELETE)
// DD *
NAME VIARDEMO(R)
/*

```

## Analyze Summary Report

Information about the analyzed program is placed in the AKR when the analyze job completes. A summary report of the run-time statistics and diagnostic messages is also produced. This report varies depending on whether you specified the SOURCE or NOSOURCE option when the analyze job was submitted.

[Figure 128](#) illustrates the Analyze Summary Report with the SOURCE option used. The information shown on this summary is described below.

**Figure 128 • Analyze Summary Report Including SmartDoc with Compile**

```

(A)
00001 000100 IDENTIFICATION DIVISION.                00010000
00002 000200 PROGRAM-ID.          VIASDDMO.           00020000
00003 000300 AUTHOR.              WRITTEN BY ASG IN LANG LEVEL 2. 00030000
000400*
00005 000500 ENVIRONMENT DIVISION.                   00050000
00006 000600 INPUT-OUTPUT SECTION.                   00060000
00007 000700 FILE-CONTROL.                            00070000
00008 000800     SELECT INFILE1  ASSIGN TO UT-S-INFILE1. 00080000
00009 000900     SELECT INFILE2  ASSIGN TO UT-S-INFILE2. 00090000
00010 001000     SELECT INFILE3  ASSIGN TO UT-S-INFILE3. 00100000

*STATISTICS*      SOURCE RECORDS = 466      DATA DIVISION STATEMENTS = 120      PROCEDURE
                   DIVISION STATEMENTS = 220
*OPTIONS IN EFFECT*  SIZE = 1048576, BUF = 262144, LINECNT = 54, SPACE1, FLAGW, SEQ
*OPTIONS IN EFFECT*  SOURCE, DMAP, PMAP, NOCLIST, SUPMAP, NOXREF, NOSXREF, LOAD, NODECK
*OPTIONS IN EFFECT*  APOST, NOTRUNC, NOFLOW, NOTERM, NONUM, NOBATCH, NONAME, COMPILE=0
*OPTIONS IN EFFECT*  NOSTATE, RESIDENT, DYNAM, LIB, NOSYNTAX, NOOPTIMIZE, NOSYMDMP
*OPTIONS IN EFFECT*  NOTEST, VERB, ZWB, SYST, NOENDJOB, NOMIGR, NOLVL, DUMP, NOADV
*OPTIONS IN EFFECT*  NOLST, NOFDECK, NOCDECK, LCOL1, L120, NOFDECK, NOCDECK, LCOL1
*OPTIONS IN EFFECT*  L120, DUMP, NOADV, NOPRINT, NOCOUNT, NOVBSUM, NOVBREF, LANGLVL(1)
(B)
ASG1534I PROGRAM VIASCOPR STARTED
ASG1519I PROGRAM VIASCOPR COMPLETED WITH RETURN CODE 0000
ASG1534I PROGRAM IKFCBL00 STARTED
ASG1519I PROGRAM IKFCBL00 COMPLETED WITH RETURN CODE 0000
ASG1534I PROGRAM VIASSYMB STARTED
ASG1519I PROGRAM VIASSYMB COMPLETED WITH RETURN CODE 0000
ASG1534I PROGRAM VIASANLZ STARTED
ASG1519I PROGRAM VIASANLZ COMPLETED WITH RETURN CODE 0000
ASG1534I PROGRAM VIADBTCH STARTED
ASG1519I PROGRAM VIADBTCH COMPLETED WITH RETURN CODE 0000
ASG1025I THE PRODUCT LEVEL FOR ASG-CENTER-OS(XA) R4.0 IS 000.
ASG1435I ASG-CENTER-OS(XA) R4.0 LVL000 -SUMMARY REPORT- PROGRAM=VIASDDMO
ASG1399I OPTIONS IN EFFECT ARE: SOURCE, NODMAP, NOPMAP.
ASG1394I SUMMARY OF OS/VIS COBOL SYMBOLS EXTRACTED FROM VIASDDMO.
ASG1395I 98 DATA NAME SYMBOLS PROCESSED.
ASG1396I 33 PROCEDURE SYMBOLS PROCESSED.
ASG1397I 131 TOTAL SYMBOLS.
ASG1398I 199 VERBS PROCESSED.
ASG1436I DIAGNOSTICS: 0 TOTAL - 0 WARNING, 0 ERROR, 0 SEVERE, 0 CATASTROPHE
ASG1437I ASG-CENTER-OS(XA) R4.0 LVL000 - END OF SYMBOL EXTRACTION FOR VIASDDMO
(C)
ASG-CENTER-OS(XA) R4.0 LVL000      PROGRAM: VIASDDMO      DD-MMM-YYYY HH:MM:SS PAGE 1
(D)
LINE  ERROR MESSAGE
      ASG0237I 131 SYMBOLS PROCESSED.
      ASG0238I 131 SYMBOLS MATCHED.
      ASG0240I 199 VERBS PROCESSED.
(E)
DIAGNOSTICS LINES: 0 TOTAL - 0 WARNINGS, 0 CONDITIONALS, 0 ERRORS, 0 DISASTERS
(F)
SOURCE LINES: 466 TOTAL - 120 DATA DIVISION STATEMENTS, 220 PROCEDURE
                   DIVISION STATEMENTS
(G)
PARAMETERS PASSED: NOCOBOLII, LANGLVL(1), FEATURES=(I,S,X)
(H)
OPTIONS IN EFFECT: BUFMAXK=2000K, FEATURES=(INSIGHT,SMARTTEST,EXTENDED), FLAG(W),
                   LINECNT=60, NORECUR, NOSEQ, NOSOURCE, SPACE1, LANGLVL(1),
(I)
ENTRY POINTS: VIASDDMO
(J)
EXTERNAL CALLS: VIASUB
(K)
END OF PROCESSING: DD-MMM-YYYY HH:MM:SS

```

## Analyze Options

These are the analyze options highlighted in [Figure 128 on page 353](#).

Option	Description
(A)	<p>A complete listing of the program is produced and shows statement numbers generated by the analyze job in the first six columns. These numbers are referenced in diagnostic messages. These notations can also appear on the source listing:</p> <p>C Statement was inserted with a COPY statement.</p> <p>** Original source statement number is out of sequence.</p> <p>I Statement was inserted with an INSERT statement.</p>
(B)	<p>This portion of the Analyze Summary is the report from the ESW monitor facility. The job steps that were executed by the monitor facility are listed along with the return codes produced.</p>
(C)	<p>The Center (Analyze) release and product level is shown along with the date and time the analysis was performed.</p>
(D)	<p>LINE and ERROR MESSAGE - This information is shown only if there were error conditions encountered. If so, this area lists the line number where the error occurred and the error message.</p>
(E)	<p>DIAGNOSTICS LINES - Indicates the total number of messages issued with subtotals for each type.</p>
(F)	<p>SOURCE LINES - Indicates the number of source lines in the program. The number of statements within the DATA DIVISION and PROCEDURE DIVISION are also shown. Each level number is counted as one statement in the DATA DIVISION. Each verb is counted as one statement in the PROCEDURE DIVISION.</p>
(G)	<p>PARAMETERS PASSED - Lists all of the analyze options specified for this analyze job.</p>
(H)	<p>OPTIONS IN EFFECT - Lists all options in effect, including default and user-specified options.</p>
(I)	<p>ENTRY POINTS - Lists the entry points in this program.</p>
(J)	<p>EXTERNAL CALLS - Lists the programs that this program CALLs.</p>
(K)	<p>END OF PROCESSING - Lists the day, month, year, and time the analyze job completed. This date and time is also reflected in the online AKR statistics.</p>

The analyze job uses many of the same options as the IBM OS/VS COBOL and the COBOL II compilers. These compile-time options are available to control the output format and to describe COBOL options. Default options are established when Insight is installed. To override the installation options, type the desired options on the Analyze Submit pop-up. If you enter an invalid option, the analyze job ignores it. If you enter a valid option more than once, the last one is processed.

Options that accept program names as parameters, with the exception of PROGRAM, accept special characters to signify generic names. The asterisk (\*) represents one or more characters. The question mark (?) represents a single character. For example:

Example	Description
DBA*	Accepts all programs that begin with DBA and end with any other characters.
D?A*	Accepts all programs that begin with D followed by any character, followed by A, then followed by any other characters.
DBA???	Accepts all programs that begin with DBA and end with any three characters.

This table summarizes the analyze options. In this summary, the abbreviations are shown in uppercase (abbreviations comply with compiler standards). These defaults reflect the information about the installation tape. The Analyze Summary Report printed at the end of each analyze job lists the actual options in effect, and the options that were passed to it (the override options).

Option	Description
BUF( <i>nnnnn</i> K) BUF= <i>nnnnn</i> K	BUF is used only as an override. Insight dynamically allocates the amount of main storage to buffers and internal tables. If an override is necessary, the minimum BUF value is 20K; the maximum BUF value is 20000K.
COBOLII COB2R3 NOCOBOLII	COBOLII and COB2R3 override the LANGLVL option and processes the input program as COBOL II. NOCOBOLII overrides the LANGLVL option and processes the input program as VS COBOL. The default is NOCOBOLII.
DB2LIB= <i>xxxxxxx</i> . <i>xxxxxx</i> . <i>xx</i> <i>xxxx</i>	DB2LIB specifies the dataset name of the load library that is used to invoke the DB2 pre-processor at your site.

Option	Description
DB2PLAN=xxxxxxxx	DB2PLAN specifies the ESW application plan that was created at installation time by the VIASBIND job. You can use DB2PLAN to override the default plan name.
DYNcall	The minimum abbreviation is DYN or NODYN.
NODYNcall	<p>DYNCALL specifies whether Insight functions use the variable name in dynamic calls as the name of the called program. If NODYNCALL is specified, dynamic calls are not processed by the analyze and information for them is not available to ESW product functions. The default is DYNCALL.</p> <p>For example, the analyze process for this code proceeds differently, depending on whether DYNCALL is specified:</p> <pre> 77 MYPROG PIC X(8). CALL MYPROG USING PARM1, PARM2. </pre> <p>In this example, if DYNCALL is in effect, the analyze process assumes that the program being called is MYPROG, regardless of the data value that MYPROG contains at run-time. The analyze process looks up the analysis results of MYPROG in the AKR to determine whether PARM1 and PARM2 are used or modified.</p> <p>If NODYNCALL is in effect for this example, the analyze process assumes that the program being called could be anything, and treats both PARM1 and PARM2 as used and modified on the call statement.</p> <p><b>Note:</b> _____  The DYNCALL option is unrelated to the COBOL compiler option DYNAM</p>

Option	Description
fLAGW fLAGE fLAG(x)	<p data-bbox="800 338 1435 569">FLAG specifies the types of messages to be listed for the analyze job. FLAGW indicates all warning and diagnostic messages are listed. FLAGE indicates diagnostic messages are listed; all other messages are suppressed. FLAG(x) indicates all messages of the specified level or above are listed. These are the valid message levels:</p> <ul data-bbox="800 579 1435 831" style="list-style-type: none"> <li data-bbox="800 579 1435 621">• I - Informational</li> <li data-bbox="800 632 1435 674">• W - Warning</li> <li data-bbox="800 684 1435 726">• E - Error</li> <li data-bbox="800 737 1435 779">• S - Severe</li> <li data-bbox="800 789 1435 831">• U - Unrecoverable</li> </ul> <p data-bbox="800 863 1435 989"><b>Note:</b> _____ Some informational messages are produced regardless of the flag setting.</p>
Input(x, x, . . . x) Input=x  NOInput(x, x, . . . x) NOInput=x	<p data-bbox="800 1020 1435 1220">INPUT lists the CALLED programs that contain INPUT statements. When commands that search for INPUT are issued, statements that CALL these programs are shown in the command results. The specified programs are in addition to those specified at installation time.</p> <p data-bbox="800 1230 1435 1367">NOINPUT overrides the installation default list of CALLED programs that contain INPUT statements. The specified programs are deleted from the default list.</p>
IO(x, x, . . . x) IO=x NOIO	<p data-bbox="800 1388 1435 1587">IO lists the CALLED programs that contain INPUT and OUTPUT statements. When commands that search for INPUT and OUTPUT are issued, statements that CALL these programs are shown in the command results. The specified programs are in addition to those specified at installation time.</p> <p data-bbox="800 1598 1435 1740">NOIO overrides the installation default list of CALLED programs that contain INPUT and OUTPUT statements. The specified programs are deleted from the default list.</p>

Option	Description
LANGLVL(1 2)	LANGLVL specifies whether to use the 1968 or 1974 American National Standard COBOL definitions when analyzing source elements with meanings that have changed. LANTLRVL(1) indicates the 1968 standard is to be used; LANTLRVL(2) indicates the 1974 standard (X3.23-1974) is to be used. The default is LANTLRVL(2).
lineCNT=60	LINECNT indicates the number of lines to be printed on each page of the source listing. This value must be 1 to 99. The default value is 60.
MAIN	MAIN is used only as an override. The EXIT PROGRAM statements in COBOL programs are treated as GOBACKs by the analyze job. This is because the program is treated as a CALLED subprogram. If the program is the main program, using the MAIN option treats the EXIT program as a fallthrough.
MBRERCNT=4000	Specifies the maximum number of analysis errors allowed for a member during the analyze job. If this number of errors is exceeded, the analyze terminates processing for that member. The number specified must be between 1 and 4000. The default is set at installation.
Output(x, x, . . . x) Output=x	OUTPUT lists the CALLED programs that contain OUTPUT statements. When commands that search for OUTPUT are issued, statements that CALL these programs are shown in the command results. The specified programs are in addition to those specified at installation time.
NOOutput(x, x, . . . x) NOOutput=x	NOOUTPUT overrides the installation default list of CALLED programs that contain OUTPUT statements. The specified programs are deleted from the default list.
PROgram(xxxxxxxxxx) PGM=xxxxxxxxxx	Analyzed programs are stored in the AKR and identified by the program name coded in the PROGRAM-ID statement. PROGRAM is used to override the name coded in the PROGRAM-ID statement. The specified name can be from one to ten characters in length.

Option	Description
RECur NORECur	RECUR specifies whether the recursion report should be included in the Analyze Summary Report. If you specify RECUR and no recursion is found, a message is issued that indicates no recursion was detected. If RECUR is specified and recursion is found, a message is issued and the recursive code is printed on the report. The default is NORECUR.
RETurn( <i>x, x, . . . x</i> ) RETurn= <i>x</i>	RETURN overrides the installation list of programs or entry points that do not return when CALLED. You override the system defaults by listing the desired programs or entry points that are to return when CALLED.
NORETurn( <i>x, x, . . . x</i> ) NORETurn= <i>x</i>	NORETURN lists the additional programs or entry points that are not to return when CALLED. When any of these programs are CALLED by the program being analyzed, they are treated as non-returning CALLS. The specified programs are in addition to the system defaults for programs that do not return when CALLED.
SEQ NOSEQ	SEQ specifies whether the analyze job is to check the source module statement number sequence. A warning message is printed if the statements are not in sequence. If the SOURCE option is specified, a flag (**) is placed between the analyze job sequence numbers and the source sequence numbers. The default is SEQ.
SOURce NOSOURce	SOURCE specifies whether the source program is to be listed. The SOURCE option is specified if a full program listing is desired at analyze time. The default is NOSOURCE.
spACE1 2 3	SPACE specifies the spacing for the source listing that is generated when you use the SOURCE option. SPACE2 causes one blank line to be placed between every source line. SPACE3 causes two blank lines to be placed between every source line. Single spacing is the default.
SQLID= <i>nnnnnnnn</i> SQLID( <i>nnnnnnnn, nnnnnnn</i> <i>n, nnnnnnnn</i> )	SQLID specifies the authorization ID or owner name (a maximum of 8 characters) that is used by the analyze process to qualify unqualified table and view references in your program.

Option	Description
SUBSYS=xxxx	SUBSYS specifies the name of the subsystem or location that designates the DBMS where the tables accessed by a specified program are stored. SUBSYS overrides the name provided at installation time.
XLIVE	<p>XLIVE is used only as an override and should only be used with programs that contain 'live' exits. Live exits are exits from perform ranges that are left dangling by imbedded PERFORMs or GO TOs in the original performed paragraph. If you do not use XLIVE, code that is unprocessed because of the live exit is ignored. If you use XLIVE, unprocessed code is saved.</p> <p>Using XLIVE can significantly increase resource usage.</p>
XMEM	XMEM is used only as an override. If a program is extremely large (i.e., 30,000 source lines) and there is insufficient memory, increase the region space. If there is still insufficient memory, type XMEM. This results in more disk I/O and additional CPU usage, but less memory consumption.

---

# 12

## AKR Utilities

---

This chapter describes AKRs and contains these sections:

Topic	Page
<a href="#">Introduction to AKR Management</a>	<a href="#">361</a>
<a href="#">AKR Structure</a>	<a href="#">362</a>
<a href="#">Online AKR Utilities</a>	<a href="#">362</a>
<a href="#">Batch AKR Utilities</a>	<a href="#">363</a>
<a href="#">Command Format</a>	<a href="#">365</a>
<a href="#">Command Syntax</a>	<a href="#">365</a>
<a href="#">Batch AKR Reports</a>	<a href="#">375</a>
<a href="#">Allocating and Expanding AKRs without ISPF</a>	<a href="#">377</a>

### Introduction to AKR Management

The AKR is the repository for all information used by ESW family of components. Analyzed programs are stored in the AKR. You can define a single AKR for use by all ESW users, or define separate AKRs for use by departments, groups, or individual users.

ESW provides both online and batch utilities for managing the AKR. This chapter describes the online AKR utilities (see "[Online AKR Utilities](#)" on page 362) and the batch AKR utilities (see "[Batch AKR Utilities](#)" on page 363).

## AKR Structure

The AKR is a BDAM or VSAM file organization. You can define an AKR to be shared by all users, or define multiple AKRs for use by departments, groups, or individuals.

**Note:** \_\_\_\_\_

See the *ASG-Center Installation Guide* for additional information about the Application Knowledge Repository.

---

## Online AKR Utilities

These are the online AKR utilities:

- File - AKR Utility pop-up that is used to rename or delete a program, or to display the AKR Directory.
- File - AKR Directory pop-up that is used to view all programs in an AKR. This pop-up can be used to rename or delete a group of programs. Statistics about the AKR are also shown on this pop-up.
- File - AKR Allocate/Expand pop-up that is used to allocate a new AKR or to expand an existing AKR.

**Note:** \_\_\_\_\_

When you use the Allocate/Expand utility, the default AKR organization type is applied. For example, if your site's default AKR type is sequential, any new AKR you allocate is created as a sequential file, and any non-sequential AKR you expand is reorganized as a sequential file.

---

## Batch AKR Utilities

The Batch AKR Utility is used to maintain the AKR without using ISPF. This table lists the commands available in the batch utility:

Command	Function
CONVERT	Converts selected members from a previous ESW product release level to the current release level.
COPY	Copies selected members from one AKR to another.
DELETE	Deletes selected members from the AKR.
EXPORT	Creates metrics and function point CDF files.
	<p><b>Note:</b> _____            EXPORT is available only to Recap users.</p>
HELP	Prints the AKR Utility Help Report.
INIT	Formats a previously-defined dataset into an AKR format.
MOVE	Copies selected members from one AKR to another and deletes them from the original AKR.
PRINT	Prints AKR directory information or COBOL source listings for selected AKR members.
PUNCH	Produces a file that contains the AKR directory information or the COBOL source code for selected AKR members.

## Job Control Statements

The Batch AKR Utility uses the JCL statements shown in [Figure 129](#). The VIAAKRIN and VIAAKROT DD statements describe AKRs that are used for AKR Utility processing. The VIASYSIN DD control cards consist of the necessary Batch AKR commands (see "[Command Format](#)" on page 365 for more information). See each command description to determine the affected DD statements.

**Figure 129 • Batch AKR JCL Statements**

```
//UTILITY EXEC PGM=VIASAKRU,REGION=3000K,PARM=''
//STEPLIB DD DISP=SHR,DSN=(ASG load library)
//VIAAKRIN DD DISP=SHR,DSN=(Input AKR)
//VIAAKROT DD DISP=SHR,DSN=(Output AKR)
//VIAPRINT DD SYSOUT=A (Print file description)
//VIAPUNCH DD SYSOUT=B (Punch file description)
//VIALOG DD SYSOUT=A (Log file)
//VIASYSIN DD *
<control cards>
//
```

## Control Cards

Commands are passed to the Batch AKR Utility with the control cards following the VIASYSIN DD statement. Control cards must conform to these standards:

- Command information must be contained in columns 1 through 72 of the control card.
- Only one command can be entered on each control card.
- Only one control card may be used per command.

All control cards with command disposition and command summaries are printed to the VIALOG AKR Utility Log file. Blank control cards are ignored.

## Command Format

Commands that use member names accept special characters to signify generic names. An asterisk (\*) represents zero or more characters. A question mark (?) represents one character. For example:

Example	Description
DBA*	Specifies all members that begin with DBA and end with any other characters.
D?A*	Specifies all members that begin with D followed by one character, followed by an A, then followed by any other characters.
DBA???	Specifies all members that begin with DBA and end with any three characters.

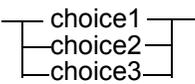
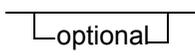
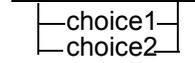
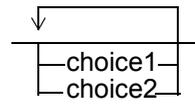
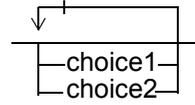
The LASTUSED parameter is used to provide the selection criteria for several commands. The specified number represents the number of days since the member was last referenced online or the date the member was analyzed if it has not been referenced.

The REPLACE parameter is used to specify that members are to be replaced on the output AKR. The NOREPLACE parameter is used to prevent members from being replaced on the output AKR. NOREPLACE is the default value.

## Command Syntax

Each Batch AKR Utility command is described in this section. The descriptions include the format and a brief explanation of the command parameters.

Item	Description
ABBREVIations	Illustrates the command abbreviation, which is shown in uppercase letters. Lowercase letters in the command are optional.
lowercase	Indicates user-supplied variable information.
UPPERCASE	Indicates commands or keywords.
<b>Bold</b>	Indicates operands that are available only if Insight is installed and a Insight analysis has been run on the COBOL program being tested.
<u>Underline</u>	Specifies the default value of an operand.

Item	Description
	Separates synonymous commands or operands.
—————>	Indicates that the command syntax is continued on the next line.
->—————	Indicates the command syntax is continued from the previous line.
—————x	Indicates the end of the command syntax.
— required —	Indicates that the operand or keyword appearing on the main command line is required.
	Indicates that one operand is required.
	Indicates that an operand or keyword appearing below the main command line is optional.
	Indicates that operands are optional.
	Indicates that more than one operand can be chosen.
	Indicates that operands can be concatenated by placing a plus sign (+) between them.

## Batch AKR Comments



### Function

Includes a comment with the commands.

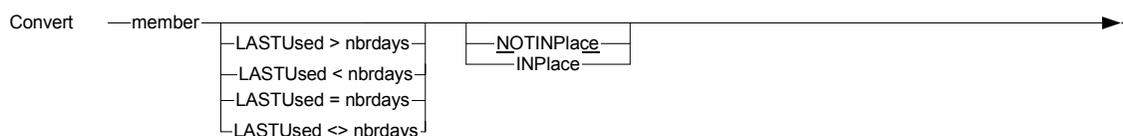
### Operands

**comment.** Indicates user-supplied text.

### Usage Notes

Blank control cards are ignored.

## CONVERT Batch AKR Command



### Function

Converts selected members that were analyzed by a prior release of ESW products to the current release level.

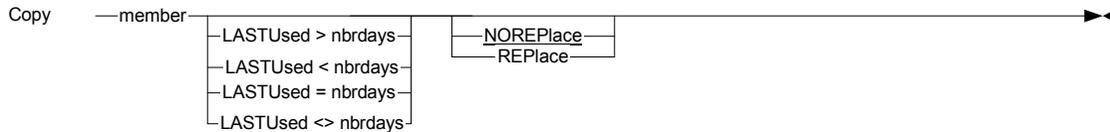
### Operands

Command	Description
<i>member</i>	Specifies a specific or generic name as described in <a href="#">"Command Format" on page 365</a> .
LASTUsed	Selects members based on the number of days since they were last used, as described in <a href="#">"Command Format" on page 365</a> .
NOTINPlace	Specifies that a member with the same name on the receiving AKR should be replaced with the member from the sending AKR.
INPlace	Specifies that a member is to be converted and kept within the AKR named in the VIAAKRIN DD statement. This option should be used with caution. Consult your systems programmer or the ASG Service Desk.

## Usage Notes

Members are copied from the AKR specified in the VIAAKRIN DD statement to the AKR specified in the VIAAKROT DD statement, as described in ["Job Control Statements" on page 364](#).

## COPY Batch AKR Command



## Function

Copies selected members from one AKR to another.

## Operands

Command	Description
<i>member</i>	Specifies a specific or generic name as described in <a href="#">"Command Format" on page 365</a> .
LASTUsed	Selects members based on the number of days since they were last used, as described in <a href="#">"Command Format" on page 365</a> .
NOTINPlace	Specifies that a member with the same name on the receiving AKR should be replaced with the member from the sending AKR.
REPlace	Replaces members that have the same name on the receiving AKR.

## Usage Notes

Members are copied from the AKR specified in the VIAAKRIN DD statement to the AKR specified in the VIAAKROT DD statement.

When you use the Allocate/Expand utility, the default AKR organization type is applied. For example, if your site's default AKR type is sequential, any new AKR you allocate is created as a sequential file, and any non-sequential AKR you expand is reorganized as a sequential file.

## DELETE Batch AKR Command

```

DELEte  —member—
          |—LASTUsed > nbrdays—
          |—LASTUsed < nbrdays—
          |—LASTUsed = nbrdays—
          |—LASTUsed <> nbrdays—
  
```

### Function

Erases selected members from the AKR.

### Operands

Operand	Description
member	Can be a specific or generic name as described in " <a href="#">Command Format</a> " on page 365.
LASTUsed	Used to select members based on the number of days since they were last used, as described in " <a href="#">Command Format</a> " on page 365.

### Usage Notes

Members are deleted from the AKR specified in the VIAAKRIN DD statement.

You cannot delete members that begin with VIA by using this command. All ESW test members begin with VIA. If these members must be deleted, use the online AKR Utility function described in "[Online AKR Utilities](#)" on page 362.

## EXPORT Batch AKR Command

EXPort -application  FPA  

### Function

Creates metrics and function point CDF files.

### Operands

Operand	Description
application	Is a specific or generic name as described in <a href="#">"Command Format" on page 365</a> .
FPA	Generates only function point information. If FPA is not specified, both metrics and function point information are generated.

### Usage Notes

EXPORT is available only for Recap users.

## HELP Batch AKR Command

HELP | ? 

### Function

Prints a description of the Batch AKR Utility and the commands that can be used.

### Operands

None.

### Usage Notes

A question mark (?) can be used as an alternate command.

The HELP command has no operands.

The Help Report is printed to the SYSOUT specified in the VIAPRINT DD statement.

## INIT Batch AKR Command



### Function

Initializes a new AKR. This internal command is used by the online AKR Utility Allocation function. See ["Online AKR Utilities" on page 362](#) for additional information.

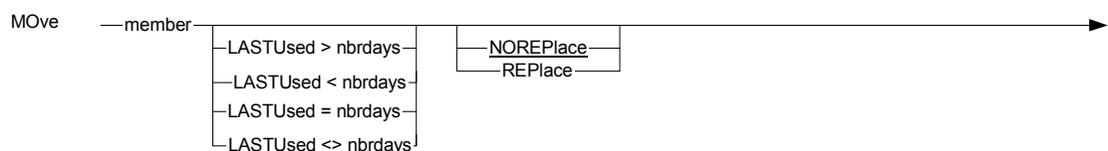
### Operand

**DSname**(*dsname*). Specifies the dataset name for the new AKR.

### Usage Notes

You must create the AKR dataset before you initialize it. The AKR that is initialized can be described in the VIAAKRIN DD statement. The VIAAKRIN DD statement is ignored if you specify the DSNAME parameter.

## MOVE Batch AKR Command



### Function

Moves selected members from one AKR to another. Specified members are copied to the receiving AKR and erased from the sending AKR.

### Operands

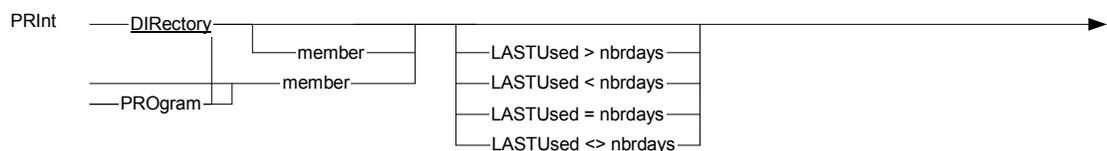
Command	Description
<i>member</i>	Specifies a specific or generic name as described in <a href="#">"Command Format" on page 365</a> .
LASTUsed	Selects members based on the number of days since they were last used, as described in <a href="#">"Command Format" on page 365</a> .

Command	Description
NOTINPlace	Specifies that a member with the same name on the receiving AKR should be replaced with the member from the sending AKR.
REPlace	Replaces members that have the same name on the receiving AKR.

### Usage Notes

Members are moved from the AKR specified in the VIAAKRIN DD statement to the AKR specified in the VIAAKROT DD statement.

### PRINT Batch AKR Command



### Function

Prints AKR directory information for the entire AKR, for a specified member, or the source code for a specified member.

### Operands

Operand	Description
Blank	If the PRINT batch AKR command is entered with no operand, the AKR directory information is printed.
DIRectory	Prints AKR directory information. This is the default. If a member is specified, the AKR directory information for that member only is printed.
PROgram	Prints the COBOL source for the specified AKR member. The generated COBOL source listing contains expansions of all COPYBOOKs and/or INCLUDEs, as well as the results of any source preprocessors such as CICS macro expansion.

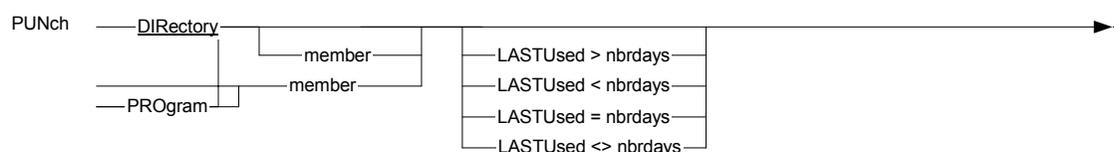
Operand	Description
<i>member</i>	Specifies a specific or generic name as described in " <a href="#">Command Format</a> " on page 365.
LASTUsed	Used to select a member based on the number of days since it was last used, as described in " <a href="#">Command Format</a> " on page 365.

### Usage Notes

Directory information or COBOL source is extracted from the AKR specified in the VIAAKRIN DD statement and is printed to the SYSOUT specified in the VIAPRINT DD statement.

See "[AKR Utility Directory Report](#)" on page 376 for more information.

### PUNCH Batch AKR Command



### Function

Produces a file that contains AKR directory information for the entire AKR, a specified member, or the source code for a specified member.

### Operands.

Operand	Description
Blank	If you enter the PRINT batch AKR command with no operand, the AKR directory information is printed.
DIRectory	Prints AKR directory information. This is the default. If you specify a member, the AKR directory information for that member only is printed.
PROgram	Prints the COBOL source for the specified AKR member. The generated COBOL source listing contains expansions of all COPYBOOKs and/or INCLUDEs, as well as the results of any source preprocessors such as CICS macro expansion.

Operand	Description
<i>member</i>	Can be a specific or generic name as described in <a href="#">"Command Format" on page 365</a> .
LASTUsed	Used to select a member based on the number of days since it was last used, as described in <a href="#">"Command Format" on page 365</a> .

### Usage Notes

Directory information or COBOL source is extracted from the AKR specified in the VIAAKRIN DD statement and is written to the file specified in the VIAPUNCH DD statement. The file that is produced is in standard IBM IEBUPDTE Utility format. ADD control cards are produced for each logical entity. The NAME parameter contains the member name for COBOL source. The NAME parameter contains AKRDIR $nn$  for directory information, where  $nn$  is a consecutively assigned number.

See the ["Punch Directory File" on page 376](#) for more information.

This table shows the format of the file produced by the PUNCH DIRECTORY command.

Description	Length	Format
Member name	10	Character
Number of source lines	6	Right justified
Days since last used	4	Right justified
Analyze date	9	DDMMYYYY
Analyze job name	8	Character
Analyze CPU	4	Character
Analyze product level	8	Character
Last reference date	9	DDMMYYYY
Last reference user ID	8	Character
Last reference CPU	4	Character

## Batch AKR Reports

Examples of the reports listed below are provided in this section.

- AKR Utility Log (see [Figure 130](#)).
- AKR Utility Directory Report (see [Figure 131 on page 376](#)).
- File produced by the PUNCH DIRECTORY command (see [Figure 132 on page 376](#)).

### AKR Utility Log

The VIALOG AKR Utility Log provides a summary of the commands issued to the Batch AKR Utility. This log contains:

- Comments
- Commands
- Completion messages
- Short summary of commands processed

The heading includes the ESW product level information, and the date and time the job was executed. Comments are enclosed in a box comprised of asterisks. The second page contains the log summary. [Figure 130 on page 375](#) illustrates the AKR Utility Log.

**Figure 130 • AKR Utility Log**

```

ASG-CENTER-OS Rx.x LVLnnn          AKR UTILITY LOG          DDMMYYYY HH:MM:SS Page 1

*****                                00140000
* PRODUCE A REPORT CONTAINING DIRECTORY INFORMATION FOR ALL *      00150000
* MEMBERS OF ASG.RENAISSA.AKR (VIAAKRIN) THAT HAVE          *      00160000
* NOT BEEN REFERENCED IN THE LAST 7 DAYS.                   *      00170000
*****                                00180000
*                                                                00190000
PRINT DIRECTORY * LASTUSE > 7                                  00200000

      ASG1289I 8 DIRECTORY ENTRIES SUCCESSFULLY PRINTED.

*****                                00210000
* PRODUCE A FILE CONTAINING DIRECTORY INFORMATION FOR ALL *      00220000
* MEMBERS OF ASG.RENAISSA.AKR (VIAAKRIN) THAT HAVE          *      00230000
* NOT BEEN REFERENCED IN THE LAST 7 DAYS.                   *      00240000
*****                                00250000
*                                                                00260000
*                                                                00270000
PUNCH DIRECTORY * LASTUSE > 7                                  00280000

      ASG1290I 8 DIRECTORY ENTRIES SUCCESSFULLY PUNCHED.

      ASG1314I *** END OF VIASYSIN ***

ASG-CENTER-OS Rx.x LVLnnn          AKR UTILITY LOG - SUMMARY DDMMYYYY HH:MM:SS Page2

ASG1301I          8 DIRECTORY ENTRIES PRINTED          0 FAILED.
ASG1302I          8 DIRECTORY ENTRIES PUNCHED          0 FAILED.

ASG1315I *** END OF SUMMARY REPORT ***

```



## Allocating and Expanding AKRs without ISPF

The Batch AKR Utility can allocate and expand VSAM AKRs without using ISPF. Existing Systems Workbench provides the VIASAKRA JCL to allocate an AKR, and the VIASAKRX JCL to expand an AKR. [Figure 133](#) is an example of the VIASAKRA JCL and [Figure 134 on page 378](#) is an example of the VIASAKRX JCL.

**Figure 133 • VIASAKRA JCL for a VSAM AKR**

```
//ASG JOB (),'ALLOC/INIT AKR'
/*ROUTE PRINT XXXXXXXX.XXXXXX
/*
/* *****
/* * ASG, INC.      ASG-CENTER Rx.x      MMM, YYYY *
/* *
/* * JCL PROCEDURE TO ALLOCATE AND INITIALIZE AN ASG *
/* * APPLICATION KNOWLEDGE REPOSITORY (AKR) *
/* *****
/*
//VIASIDAP PROC SYSOUT='*',      PRINTED OUTPUT MESSAGE CLASS
//      VIASOFT='ASG',      HIGH LEVEL NODE FOR ASG DATA SETS
//      CENTER='VIACENxx' MIDDLE NODE FOR ASG DATA SETS
/*
/* *****
/* * DEFINE A NEW ASG AKR FILE *
/* *****
/*
//DEFAKR EXEC PGM=IDCAMS,REGION=512K
//SYSPRINT DD SYSOUT=&SYSOUT
//SYSOUT DD SYSOUT=&SYSOUT
//SYSIN DD DDNAME=SYSIN
/*
/* *****
/* * INITIALIZE THE NEWLY ALLOCATED AKR FILE *
/* *****
/*
//INITAKR EXEC PGM=VIASAKRU,REGION=2048K,COND=(0,LT,DEFAKR)
//STEPLIB DD DSN=&ASG..&CENTER..LOADLIB,DISP=SHR
//VIASYSIN DD DDNAME=SYSIN
//VIALOG DD SYSOUT=&SYSOUT
//SYSUDUMP DD SYSOUT=&SYSOUT
/*
/* *****
/* * DELETE NEW AKR (ONLY IF INITAKR FAILS) *
/* *****
/*
//DELETE EXEC PGM=IDCAMS,REGION=512K,
//      COND=(EVEN,(0,LT,DEFAKR),(0,EQ,INITAKR))
//SYSPRINT DD SYSOUT=&SYSOUT
//SYSOUT DD SYSOUT=&SYSOUT
//SYSIN DD DDNAME=SYSIN
/*
//      PEND
/*
//VIASAKRA EXEC VIASIDAP
/*
//DEFAKR.SYSIN DD *
```

```

DEFINE CLUSTER -
  (NAME (XX) - XX AKR NAME HERE **
  XX (XX) - XX ALLOC UNITS AND QUANTITY HERE **
  VOLUME (SRT801) -
  CONTROLINTERVALSIZE (4096) -
  NUMBERED -
  RECORDSIZE (4089 4089) -
  RECOVERY -
  ERASE -
  UNIQUE -
  SHAREOPTIONS (3 3) -
  DATA -
  (NAME (XX.DATA) ) XX AKR NAME HERE **
/*
//INITAKR.VIASYSIN DD *
INIT DSNNAME (XX) XX AKR NAME HERE **
/*
//DELETE.SYSIN DD *
DELETE XX XX AKR NAME HERE **
/*

```

To allocate an AKR, replace XX in the NAME(XX), NAME(XX.DATA), and the DSNNAME(XX) parameters with the name of the AKR to be allocated. Then replace XX(XX) with the allocation units values and quantities for your site.

Figure 134 • VIASAKRX JCL for a VSAM AKR

```

//ASG JOB ( ), 'EXPAND AKR'
/* INSERT '/*ROUTE PRINT NODE.USER' HERE IF NEEDED.
/*
/* *****
/* * ASG, INC. ASG-CENTER Rxx.x MMM, YYYY *
/* * * * * *
/* * JCL PROCEDURE TO EXPAND AN EXISTING ASG *
/* * APPLICATION KNOWLEDGE REPOSITORY (AKR). *
/* *****
/*
//VIASAKXP PROC SYSOUT='*', PRINTED OUTPUT MESSAGE CLASS
// VIASOFT='ASG', HIGH LEVEL NODE FOR ASG DATA SETS
// CENTER='VIACENxx' MIDDLE NODE FOR ASG DATA SETS
/*
/* *****
/* * DEFINE A NEW DATA SET *
/* *****
/*
//DEFAKR EXEC PGM=IDCAMS,REGION=512K
//SYSPRINT DD SYSOUT=&SYSOUT
//SYSOUT DD SYSOUT=&SYSOUT
//SYSIN DD DDNAME=SYSIN
/*
/* *****
/* * INITIALIZE NEW DATA SET AS ASG AKR *
/* *****
/*
//INITAKR EXEC PGM=VIASAKRU,REGION=2048K,COND=(0,LT)
//STEPLIB DD DSN=&ASG..&CENTER..LOADLIB,DISP=SHR
//VIASYSIN DD DDNAME=SYSIN
//VIALOG DD SYSOUT=&SYSOUT
//SYSUDUMP DD SYSOUT=&SYSOUT
/*

```

```

//* *****
//* * COPY OLD AKR TO NEW AKR *
//* *****
//*
//REPRO EXEC PGM=IDCAMS,REGION=512K,COND=(0,LT)
//SYSPRINT DD SYSOUT=&SYSOUT
//SYSOUT DD SYSOUT=&SYSOUT
//SYSIN DD DDNAME=SYSIN
//*
//* *****
//* * UPDATE INTERNAL SIZE OF ASG AKR *
//* *****
//*
//RESIZE EXEC PGM=VIASAKRU,REGION=2048K,COND=(0,LT)
//STEPLIB DD DSN=&ASG..&CENTER..LOADLIB,DISP=SHR
//VIASYSIN DD DDNAME=SYSIN
//VIALOG DD SYSOUT=&SYSOUT
//SYSUDUMP DD SYSOUT=&SYSOUT
//*
//* *****
//* * DELETE OLD AKR AND RENAME NEW AKR TO *
//* * OLD AKR NAME *
//* *****
//RENAME EXEC PGM=IDCAMS,REGION=512K,COND=(0,LT)
//SYSPRINT DD SYSOUT=&SYSOUT
//SYSOUT DD SYSOUT=&SYSOUT
//SYSIN DD DDNAME=SYSIN
//*
//* *****
//* * RUN THIS STEP ONLY IF ALL ABOVE STEPS *
//* * RUN SUCCESSFULLY *
//* *****
//TESTCODE EXEC PGM=IEFBR14,COND=(0,LT)
//SYSIN DD DDNAME=SYSIN
//*
//* *****
//* * DELETE OLD AKR AND RENAME NEW AKR TO *
//* * OLD AKR NAME *
//* *****
//RENAME EXEC PGM=IDCAMS,REGION=512K,COND=(0,LT)
//SYSPRINT DD SYSOUT=&SYSOUT
//SYSOUT DD SYSOUT=&SYSOUT
//SYSIN DD DDNAME=SYSIN
//*
//* *****
//* * RUN THIS STEP ONLY IF ALL ABOVE STEPS *
//* * RUN SUCCESSFULLY *
//* *****
//TESTCODE EXEC PGM=IEFBR14,COND=(0,LT)
//SYSIN DD DDNAME=SYSIN
//*

```

```

/** *****
/** * DELETE NEW AKR ONLY IF EXPAND IS NOT *
/** * SUCCESSFUL *
/** *****
/**
//DELETE EXEC PGM=IDCAMS,REGION=512K,
// COND=(EVEN,(0,LT,DEFAKR),(0,EQ,TESTCODE))
//SYSPRINT DD SYSOUT=&SYSOUT
//SYSOUT DD SYSOUT=&SYSOUT
//SYSIN DD DDNAME=SYSIN
/**
// PEND
/**
//VIASAKRX EXEC VIASAKXP
/**
//DEFAKR.SYSIN DD *
DEFINE CLUSTER -
    (NAME (ASG.VIACENxx.AKR.EX) /* AKR NAME */ -
    RECORDS(6000) /* NEW ALLOC UNITS AND QUANTITY */ -
    VOLUME(XXXXXX) /* AKR VOLSER */ -
    CONTROLINTERVALSIZE(4096) -
    NUMBERED -
    RECORDSIZE(4089 4089) -
    RECOVERY -
    UNIQUE -
    SHAREOPTIONS(3 3) -
    DATA -
    (NAME (ASG.VIACENxx.AKR.EX.DATA)) /* AKR NAME **
/**
//INITAKR.VIASYSIN DD *
INIT DSNAME (ASG.VIACENxx.AKR.EX) /* AKR NAME **
/**
/**
//REPRO.SYSIN DD *
REPRO INDATASET (ASG.VIACENxx.AKR) /* AKR NAME */ -
    OUTDATASET (ASG.VIACENxx.AKR.EX) /* AKR NAME */ -
    REPLACE
/**
/**
//RESIZE.VIASYSIN DD *
RESIZE DSNAME (ASG.VIACENxx.AKR.EX) /* AKR NAME **
/**
/**
//RENAME.SYSIN DD *
DELETE ASG.VIACENxx.AKR /** AKR NAME HERE **/
ALTER ASG.VIACENxx.AKR.EX /* AKR NAME */ -
    NEWNAME (ASG.VIACENxx.AKR) /* AKR NAME */
ALTER ASG.VIACENxx.AKR.EX.DATA /* AKR NAME */ -
    NEWNAME (ASG.VIACENxx.AKR.DATA) /* AKR NAME */
/**
/**
//DELETE.SYSIN DD *
DELETE ASG.VIACENxx.AKR.EX /* AKR NAME HERE */
/**

```

***To expand an AKR***

- 1 Replace *xx* in the NAME(*xx*.EX), NAME(*xx*.EX.DATA), DSNAME(*xx*.EX), INDATASET(*xx*), and the OUTDATASET(*xx*.EX) parameters with the name of the AKR to be expanded.
- 2 Replace *xx(xx)* with the allocation units values and quantities for your site.

***To rename an AKR***

- 1 Replace the *xx* in the DELETE statement with the AKR to be renamed.
- 2 Type the new AKR name in the NEWNAME(*xx*) parameter.



---

# 13

## Paragraph Control Transfer Methods

---

The Paragraph Cross Reference pop-up shows how control is transferred to or from the target paragraphs. This table lists how control is transferred.

Method	Abbreviations
COBOL Transfer Methods	
ALTERED GOTO	ALTERED, AG
CALL INTERNAL	CLL INT, CI
CALL RETURN	CLL RTN, CR
FALLTHRU	FALL, F
GOTO	G
GOTO DEPENDING	GODEP, GD
PERFORM	PERF, P
PROGRAM ENTRY	ENTRY
PROGRAM EXIT	EXIT
RETURN	RET, R
UNKNOWN	UNKN, ?
USE BEFORE RPT	USE REPORT, U
USE FOR DEBUG	USE DEBUG, U
USE ON ERROR	USE ERROR, U

Method	Abbreviations
CICS Handle Strings	
ABEND	ABND, ABN
ANYKEY	ANYK, AK
CBIDERR	CBIDE, CB
CCERROR	CCERR, CC
CLEAR	CLR, CL
CLRPARTN	CLRP, CLP
DISABLED	DISA, DI
DSIDERR	DSIDE, DS
DSSTAT	DSST, DT
DUPKEY	DUPK, DK
DUPREC	DUPR, DR
ENDDATA	ENDD, ED
ENDFILE	ENDF, EF
ENDINPT	ENDI, EI
ENQBUSY	ENQB, EB
ENTER	ENT, ET
ENVDEFERR	ENVDE, EV
EOC	EC
EODS	ES
EOF	EF
ERROR	ERR, ER
EXPIRED	EXPIR, EX

Method	Abbreviations
FUNCERR	FUNCE, FE
IGREQCD	IGQCD, ICD
IGREQID	IGQID, ID
ILLOGIC	ILLOG, IL
INBFMH	INBF, IB
INVERRTERM	INVERRT, IT
INVLDC	INVL, ILD
INVMPSZ	INVM, IM
INVPARTN	INVP, IP
INVPARTNSET	INVPART, IS
INVREQ	INVR, IR
INVTSREQ	INVTS, IT
IOERR	IOE, IE
ISCINVREQ	ISCIR, IIR
ITEMERR	ITEME, ITE
JIDERR	JIDE, JID
LENGERR	LENGE, LE
LIGHTPEN	LPEN, LPN
MAPERROR	MAPE, MPE
MAPFAIL	MAPF, MPF
NAMEERROR	NAMEE, NME
NOJBUFSP	NOJBSP, NJB
NONVAL	NONV, NV

Method	Abbreviations
NOPASSBKRD	NOPASSB, NPB
NOPASSBKWR	NOPASSBW, NPW
NOSPACE	NOSA, NSA
NOSPOOL	NOSP, NSP
NOSTART	NOST, NST
NOSTG	NSTG, NTG
NOTALLOC	NALLC, NAL
NOTAUTH	NAUTH, NAH
NOTFND	NFND, NFD
NOTOPEN	NOPEN, NOP
OPENERR	OPENE, OPE
OVERFLOW	OVRFLW, OVF
OPERID	OPEI, OPI
PA1	A1, 1
PA2	A2, 2
PA3	A3, 3
PARTNFAIL	PARTNF, PNF
PF1	F1
PF2	F2
PF3	F3
PF4	F4
PF5	F5
PF6	F6

Method	Abbreviations
PF7	F7
PF8	F8
PF9	F9
PF10	F10
PF11	F11
PF12	F12
PF13	F13
PF14	F14
PF15	F15
PF16	F16
PF17	F17
PF18	F18
PF19	F19
PF20	F20
PF21	F21
PF22	F22
PF23	F23
PF24	F24
PGMIDERR	PGMIDE, PDE
QBUSY	QBSY, QB
QIDERR	QIDE, QID
QZERO	QZRO, QZ
RDATT	RDAT, RDT

Method	Abbreviations
RETPAGE	RETP, RTP
ROLLEDBACK	ROBACK, RBK
RTEFAIL	RTEF, RTF
RTESOME	RTES, RTS
SELNERR	SELNE, SNE
SESSBUSY	SESSB, SSB
SESSIONERR	SESSIONE, SSE
SIGNAL	SIGNL, SGL
SUPPRESSED	SUPPRESS, SPS
SYSBUSY	SYSB, SYB
SYSIDERR	SYSIDE, SID
TERMERR	TERME, TME
TERMIDERR	TERMIDE, TID
TRANSIDERR	TRANSIDE, TSI
TRIGGER	TRIG, TRG
TSIOERR	TSIOE, TIO
UNEXPIN	UNEXP, UNP
USERIDERR	USERIDE, UID
WRBRK	WRBK, WBK
WRONGSTAT	WRONGS, WST

Method	Abbreviations
SQL WHENEVER Statements	
SQLCONTINUE	SQLCONT, SQC
SQLWARNING	SQLWARN, SQW
SQLERROR	SQLE, SQE
SQLNOTFOUND	SQLNF, SQN



---

# 14

## Help Facility

---

This chapter describes Insight Help options and contains these sections:

Topic	Page
<a href="#">Help Navigational Commands</a>	<a href="#">393</a>
<a href="#">Screen and Pop-up Help</a>	<a href="#">394</a>
<a href="#">Command Help</a>	<a href="#">395</a>
<a href="#">General Information</a>	<a href="#">397</a>
<a href="#">Specific Information</a>	<a href="#">397</a>
<a href="#">Help Abends</a>	<a href="#">398</a>
<a href="#">Help Messages</a>	<a href="#">399</a>

Alliance provides a comprehensive and context sensitive Help facility, including an online Help Tutorial, to answer most questions online. The Help Tutorial contains help information on topics such as pull-downs, screens, pop-ups, commands, messages and abends. The Help Tutorial also includes a Table of Contents that lists general information topics, and a comprehensive Index for viewing specific information.

You can access online help using these methods:

- Select Help on the action bar to display the help pull-down and press Enter.
- Press PF01/PF13 on any Insight screen.
- Type `HELP`.
- Type a question mark (?) in the command input area on any screen or pop-up and press Enter.

This table lists the Insight online help information, and the means of accessing them.

Help Topic	How to Access
Screen and pop-up help	Request help for the current screen or pop-up by selecting the Current screen action on the Help pull-down, by typing <code>HELP</code> , or by pressing PF01/13. No messages can appear on the screen or pop-up at the same time as when you're requesting help.
Command help	Request help for a command by typing the command in the command input area and pressing PF01/13, typing <code>HELP COMMANDS</code> , typing <code>HELP</code> plus the command name, selecting All Commands on the Help pull-down, or by selecting Help ► specific Command.
General information	Request general help information by selecting the Help ► Table of contents or by typing <code>TOC</code> from within the Help Tutorial.
Specific information	Request help for specific topics by selecting Help ► Index, selecting option 6 on the Help Table of Contents or by typing <code>INDEX</code> from within the Help Tutorial. View help for a specific topic by selecting the appropriate index entry.
Abends	Request help for ESW user abends by selecting the Common abends action on the Help pull-down or by typing <code>HELP ABENDS</code> . The Help Tutorial Abends screen displays. Select topic 2 on this screen to display the ESW Abend Codes screen, that lists all the ESW user abends, and explanations for each abend.
Messages	Display help for a current message by selecting the current message action on the Help pull-down. Request help for a short message, displayed in the upper right corner of the screen, by typing <code>HELP</code> or pressing PF01/13. The corresponding long message displays. Display help for a specific message by selecting Help ► Specific message, or by typing <code>HELP msg#</code> .

## Help Navigational Commands

All online help topics listed in the previous table are contained in the Help Tutorial. To reach an online help topic from anywhere within the Help Tutorial, go through the Help Table of Contents or Index. Insight provides commands for navigating the Help Tutorial. These commands are listed below.

Command	Purpose
BACK	Redisplays the previous Help Tutorial screen.
END	Exits the Help Tutorial.
ENTER key	Displays the next screen in a continuation series.
INDEX	Displays the first screen of the Help Index.
SKIP	Goes directly to the next subject.
TOC	Displays the Help Table of Contents.
UP	Displays the next higher-level subject.
Alpha character	On an Index screen, entering an alphabetic character displays the Index screen corresponding to that character.

## Screen and Pop-up Help

The Help Tutorial for each screen and pop-up describes all the options available on that screen or pop-up, lists descriptions of all the fields, and notes any special processing considerations.

To request Help for the current screen or pop-up, follow this step:

- ▶ Select Help ▶ Current screen, type HELP or HELP SCREEN, or press PF1/13 with no messages appearing on the screen. [Figure 135](#) illustrates the Help Tutorial for the Options - Product Allocations pop-up.

**Figure 135 • Screen and Pop-up Help Example**

```
ASG-Insight Rx.x ----- Options - Product Allocations ----- HELP
===>
The Options - Product Allocations pop-up is used to specify the allocation
parameters for the Log, List, Punch, and Work files. This pop-up is
displayed by selecting the Allocation... action on the Options pull-down, or
by entering the ALLOCDEF command on any screen.

Note: Management Class, Storage Class, and Data Class provide various
parameters for newly allocated data sets. These parameters apply only
if you have SMS active at your site. Your system administrator
determines the valid entries for these parameters.

The Storage Class and Volume serial parameters are mutually exclusive.

Field Descriptions

Log File      Specify either the Management Class and Storage Class or the
              Generic unit and volume serial number for the Log file
              that is allocated upon entry into ASG-Insight. The Log file
              is used for error messages and log commands. File characteristics
              are specified on the Options - Log/List/Punch Definition pop-up.

              (continued)
```

## Command Help

To request Help for a specific command, select one of these actions:

- Select Help ► Specific and press Enter.
- Type the command in the command input area and press Enter.
- Press PF1/13.
- Type HELP followed by the desired command name and press Enter.

A message displays describing the command. Press PF1/13 to display the Help Tutorial screen for that command.

The Help Tutorial for each command displays the command syntax diagram, and gives a description of each operand in the command.

To display a list of all Insight primary commands, type UP on a command help screen.

For help on all Insight commands, follow this step:

- Select Help ► All and press Enter.

**Or**

Type `HELP COMMANDS` and press Enter.

The All commands action displays a complete list of all Insight commands.

The `HELP COMMANDS` command displays a message that lists most of the primary commands. After this message displays, press PF1/13 to display the list of all Insight commands. From this list, select the appropriate number to display information about a particular command.

[Figure 136](#) illustrates the Help Tutorial screen for the RECALL command.

**Figure 136 • Command Help Example**

```

ASG-Insight Rx.x ----- RECALL ----- HELP
===>

The RECALL command displays the previous command or message. The last twenty
commands that have been executed and the last twenty messages that have been
displayed are stacked. RECALL can be entered repeatedly to display any of the
stacked commands or messages. Once the desired command displays, it can be
executed again by pressing ENTER or changed prior to execution.

The RECALL command syntax is:

RECall
-----><
      |-COMmand|CMD-|      |-NEXT-|
      |-MESSage|MSG-|      |-PREV-|
      |-POPup-----|
                                         Minimum Abbreviations are in CAPS
                                         Default operands are highlighted
LEGEND: ---required-----><
                                         |-optional-|

The following topic will be presented only if explicitly selected by number:

  1 - Operand Descriptions
    
```

For some commands, (on the Source View screen only), pressing PF1/13 after the long message displays causes NOTES to display with specific examples for using the command. After the NOTES are displayed, pressing PF1/13 again displays the Help Tutorial for that command. The Help NOTES displays for those commands where specific examples are helpful.

[Figure 137](#) illustrates the Help NOTES for the RECALL command.

**Figure 137 • Help NOTES Example**

```

File View Search Logic Task List Options Help
-----
Command ===> Source View Program: VIAIDEMO
Scroll ===> CSR
ASG4541I RECALL REDISPLAYS THE PREVIOUS COMMAND OR MESSAGE.
=NOTE +----- Examples ----- RECALL ----- Descriptions -----+
=NOTE | RECALL | | Recalls the last command that was |
=NOTE | | | entered in the primary command |
=NOTE | | | area. |
=NOTE | REC MSG | | Recalls the last message that was |
=NOTE | | | displayed in the message area. |
=NOTE | RECALL NEXT | | Once recalling is started, NEXT |
=NOTE | | | may be used to reverse its |
=NOTE | | | direction. |
=NOTE +-----+
***** ***** TOP OF DATA *****
000001 IDENTIFICATION DIVISION.
000002 PROGRAM-ID. VIAIDEMO.
000003 AUTHOR. WRITTEN BY ASG AT LANGLVL 2.
000004 ENVIRONMENT DIVISION.
000005 CONFIGURATION SECTION.
000006 SOURCE-COMPUTER. IBM-370.
000007 OBJECT-COMPUTER. IBM-370.
000008 INPUT-OUTPUT SECTION.
000009 FILE-CONTROL.
000010 SELECT MASTERIN ASSIGN TO S-MASTERIN.
    
```

## General Information

To request general help information, follow this step:

- ▶ Select Help ▶ Table of contents or type TOC on any Help Tutorial screen and press Enter. The Help Table of Contents, shown in [Figure 138](#), displays.

**Figure 138 • Help Table of Contents**

```

ASG-Insight Rx.x ----- Help Table of Contents ----- HELP
===>

The topics below represent general categories of information about
ASG-Insight.
To get help for a pull-down, select the Action Bar topic. This Help Table of
Contents may be redisplayed from any Help screen by entering the TOC command.

The following topics will be presented only if explicitly selected by number:

  1 Overview of ASG-Insight
  2 Introduction to CUA
  3 The Action Bar
  4 Customer Support
  5 Release x.x Summary of Revisions
  6 Index for ASG-Insight Help

```

## Specific Information

To request Help for specific topics, choose one of these actions:

- Select Help ▶ Index and press Enter.
- Select option 6 on the Help Table of Contents and press Enter.
- Type INDEX from within the Help Tutorial and press Enter.

Help for a specific topic can then be viewed by selecting the appropriate Index entry.

On any Index screen, enter an alphabetic character to display the Index screen corresponding to that character.

[Figure 139](#) illustrates a Help Index screen.

**Figure 139 • Help Index Example**

```
ASG-Insight Rx.x ----- INDEX A - C ----- HELP
===>

To select a topic, enter the two- or three-character identifier.

A1 - Action Bar                B1 - BRANCH Command
A2 - AKR                       B2 - Branch Previous Options
A3 - AKR Program Selection     B3 - Buf, Analyze Option
A4 - ALLOCDEF Command         B4 - BYPASS PERFORMED CODE,
A5 - ANALYZE Command          Decision Option
A6 - Analyze Options
A7 - At End, Decision Option   C1 - CALL Command
A8 - Automated Task - Debug a  C2 - CANCEL Command
    Program Abend              C3 - COBOL Language Subset
A9 - Automated Task - Modify a C4 - COBOL II, Analyze Option
    Computation                C5 - Commands
A10 - Automated Task - Modify Program C6 - COPY Command
    Output                      C7 - CUA, Introduction
A11 - Automated Task - Result Options C8 - CURRENT Command
                                C9 - Current cursor location

Another index page can be displayed by entering its letter.
```

## Help Abends

To request Help for user abends, choose one of these options:

- Select Help ► Common abends and press Enter.
- Type HELP ABENDS and press Enter.
- Type ABENDS in the command input area and press PF1/13.

The ABENDS screen displays. Select topic 2 on this screen to display the ASG Abend Codes screen, that lists all the user abend messages, and explanations for each message.

[Figure 140](#) illustrates the ASG Abend Codes screen.

**Figure 140 • ASG Abend Code Screen**

```

ASG-Insight Rx.x ----- ASG ABEND CODES ----- HELP
==>

Abend codes in the range 900 - 999 (X'384 - X'3E7') bypass ASG error
recovery, causing the abend to be handled by ISPF or by the system. If the
problem cannot be resolved, call Customer Support.

965 X'3C5'    Unable to intercept program.
968 X'3C8'    An internal error occurred during initialization.
970 X'3CA'    A package load module was called directly.
972 X'3CC'    The ASG Edit Monitor encountered a severe error.
974 X'3CE'    An invalid VIASPRMS module was found. The current product
              expects a level of CEO30 or greater. Enter HELP 4988 for more
              information.

                               (continued)

```

## Help Messages

Insight messages are displayed in the long message area. This is the format for help messages:

```
ASGnnnnx text
```

where:

*nnnn* is the message number.

*x* is one of the severity levels listed below.

*text* is the long or short message text.

Level	Type	Description
I	Informational	Specifies that no required action.
W	Warning	Specifies that an error condition exists that is not critical
E	Error	Specifies that a critical error condition exists
D	Disaster	Specifies that a serious error condition exists; the product is unable to continue
T	Termination	Specifies that the product terminated with the specified error.

Short messages are displayed when available. Long messages are displayed if a short message does not exist or when help is requested immediately following a displayed short message.

To display Help for a specific message, follow this step:

- ▶ Select Help ▶ Current message or Help ▶ Specific message and press Enter.

**Or**

Type HELP followed by the message number and press Enter.

The Help Explanation and Action screen for that message displays. [Figure 141](#) illustrates the Help Explanation and Action screen.

**Figure 141 • Help Explanation and Action Screen**

```
ASG-Insight - Rx.x --- HELP EXPLANATION AND ACTION PANEL ----- HELP
===>                                                                    SCROLL ===> CSR

ASG0650  END OF PROGRAM; USE BRANCH BACKUP TO FOLLOW OTHER BRANCHES.

EXPLANATION:
  This is a warning message indicating that branch has reached the
  physical end of the program.

ACTION:
  If you wish to branch to other paths, use BRANCH BACKUP to reach
  the desired decision point and then use BRANCH to follow another
  path.

***** BOTTOM OF DATA *****
```

## **Printing Messages**

You can either print all Insight messages or a range of messages using the VIASMPRT program. The VIASMPRT program produces a listing of the specified messages that includes:

- Message number
- Short message (if available)
- Long message
- Explanation of the message
- Action (if any)

JCL to execute the VIASMPRT program is in ASG.VIACENxx.CNTL(VIASMPRT). The entire message file is printed unless a you specify a range in the PRM parameter. For example: this command prints messages 300 through 499.

```
PRM=' START=300,END=499 '
```

To print all messages, specify the ALL keyword in the PRM parameter.

The default value for START is 1; the default value for END is 5000. If you enter only the START value, messages print starting at the message number you specified and ending with 5000. If you enter only the END value, messages start printing with 1 and end with the message number you specified.

The NOTES keyword specifies that any notes associated with a message are printed. The default is NONOTES. Typically, notes are provided to illustrate Center primary commands.

[Figure 142](#) and [Figure 143 on page 402](#) illustrate the VIASMPRT JCL and the output from the job.

**Figure 142 • VIASMPRT JCL**

```
//ASG      JOB ( ), 'ASG-CENTER VIASMPRT'
//*      INSERT /*ROUTE PRINTNODE.USER' HERE IF NEEDED.
//*
//*      *****
//*      * ASG, INC.          ASG-CENTER Rx.x          DEC, 2001      *
//*      *
//*      *          UTILITY TO PRINT ASG MESSAGES          *
//*      *
//*      *****
//*
//VIASMPRT PROC VIASOFT='ASG',          HIGH LEVEL NODE OF ASG DATA SETS
//          CENTER='VIACENxx',          MIDDLE NODE OF ASG DATA SETS
//          SYSOUT='*',          PRINT OUTPUT MESSAGE CLASS
//          PRM=''          PARM FOR MESSAGES TO BE PRINTED
//*
//*      *****
//*      *
//*      *          M E S S A G E   P R I N T   U T I L I T Y          *
//*      *
//*      * THIS PROGRAM WILL PRINT ALL OF THE MESSAGES IN THE ASG      *
//*      * MESSAGE FILE AND THE HELP TEXT ASSOCIATED WITH EACH        *
//*      * MESSAGE IT WILL PRINT THE ENTIRE FILE BY DEFAULT. YOU MAY  *
//*      * SELECT A GIVEN RANGE OF MESSAGES BY SPECIFYING THE OPTION-  *
//*      * AL PARAMETER KEYWORDS: START AND END. FOR EXAMPLE:        *
//*      * PRM='START=300,END=499'                                     *
//*      * WILL PRINT MESSAGES NUMBER 300 THROUGH 499, INCLUSIVE.    *
//*      * THE DEFAULT VALUES FOR START AND END ARE 1 AND 99999      *
//*      * RESPECTIVELY. CONSEQUENTLY THE PRM VALUE 'END=300' WILL    *
//*      * PRINT MESSAGES 1 THROUGH 300, AND THE PRM VALUE           *
//*      * 'START=4000' WILL PRINT MESSAGES 4000 THROUGH 99999.     *
//*      *
//*      * AN OPTIONAL KEYWORD, NOTES, WILL ALSO PRINT ANY NOTES    *
//*      * ASSOCIATED WITH A MESSAGE.                                  *
//*      *
//*      * ADDITIONALLY, THE KEYWORD 'ALL' WILL EXPLICITLY PRINT ALL  *
//*      * MESSAGES.                                                  *
//*      * *****
//*
//*
//VIAMPRT EXEC PGM=VIASMPRT,REGION=4096K,
//          PARM='&PRM'
//STEPLIB DD DSN=&ASG..&CENTER..LOADLIB,DISP=SHR
//VIAMSGS DD DSN=&ASG..&CENTER..VIAMSGS,DISP=SHR
//SYSPRINT DD SYSOUT=&SYSOUT
//VIAPRINT DD SYSOUT=&SYSOUT
//VIALOG DD SYSOUT=&SYSOUT
//SYSUDUMP DD SYSOUT=&SYSOUT
//*
//          PEND
//*
//VIASMPRT EXEC VIASMPRT          PRINT MESSAGES
//*
```

Figure 143 • VIASMPRT Output

```

//ASG      JOB ( ),'ASG-CENTER VIASMPRT'
//*      INSERT '/*ROUTE PRINTNODE.USER' HERE IF NEEDED.
//*
//*      *****
//*      * ASG, INC.          ASG-CENTER  R6.0          DEC, 2001  *
//*      *
//*      *          UTILITY TO PRINT ASG MESSAGES          *
//*      *
//*      *****
//*
//VIASMPRT PROC VIASOFT='ASG', HIGH LEVEL NODE OF ASG DATA SETS
//          CENTER='VIACENxx', MIDDLE NODE OF ASG DATA SETS
//          SYSOUT='*', PRINT OUTPUT MESSAGE CLASS
//          PRM=''          PARM FOR MESSAGES TO BE PRINTED
//*
//*      *****
//*      *          M E S S A G E   P R I N T   U T I L I T Y          *
//*      *
//*      * THIS PROGRAM WILL PRINT ALL OF THE MESSAGES IN THE ASG *
//*      * MESSAGE FILE AND THE HELP TEXT ASSOCIATED WITH EACH *
//*      * MESSAGE IT WILL PRINT THE ENTIRE FILE BY DEFAULT. YOU MAY *
//*      * SELECT A GIVEN RANGE OF MESSAGES BY SPECIFYING THE OPTION- *
//*      * AL PARAMETER KEYWORDS: START AND END. FOR EXAMPLE: *
//*      * PRM='START=300,END=499' *
//*      * WILL PRINT MESSAGES NUMBER 300 THROUGH 499, INCLUSIVE. *
//*      * THE DEFAULT VALUES FOR START AND END ARE 1 AND 99999 *
//*      * RESPECTIVELY. CONSEQUENTLY THE PRM VALUE 'END=300' WILL *
//*      * PRINT MESSAGES 1 THROUGH 300, AND THE PRM VALUE *
//*      * 'START=4000' WILL PRINT MESSAGES 4000 THROUGH 99999. *
//*      *
//*      * AN OPTIONAL KEYWORD, NOTES, WILL ALSO PRINT ANY NOTES *
//*      * ASSOCIATED WITH A MESSAGE. *
//*      *
//*      * ADDITIONALLY, THE KEYWORD 'ALL' WILL EXPLICITLY PRINT ALL *
//*      * MESSAGES. *
//*      *****
//*
//*
//VIAMPRT EXEC PGM=VIASMPRT,REGION=4096K,
//          PARM='&PRM'
//STEPLIB DD DSN=&ASG..&CENTER..LOADLIB,DISP=SHR
//VIAMSGS DD DSN=&ASG..&CENTER..VIAMSGS,DISP=SHR
//SYSPRINT DD SYSOUT=&SYSOUT
//VIAPRINT DD SYSOUT=&SYSOUT
//VIALOG DD SYSOUT=&SYSOUT
//SYSUDUMP DD SYSOUT=&SYSOUT
//*
//          PEND
//*
//VIASMPRT EXEC VIASMPRT          PRINT MESSAGES
//*

```

---

## Glossary

---

### **action bar**

The line of keywords at the top of a screen. Each keyword represents a category of actions that may be performed on that screen. An action is selected by moving the cursor to the desired keyword and pressing Enter. See the *ASG-Insight User's Guide* for additional information.

### **AKR**

See "[Application Knowledge Repository](#)" on page 404 .

### **AKR utility log**

Provides a summary of the commands issued to the Batch AKR Utility. See "[AKR Utilities](#)" on page 361 for more information.

### **alias of**

A field on a pop-up listing entries in the AKR. If the analyzed program contains an ENTRY point, Alias Of is the name of the program that contains the ENTRY point. If the name in the PROGRAM-ID statement was overridden at the time the analyze job was submitted, Alias Of is the name that was entered in the AKR program name field on the File - Analyze Submit pop-up.

### **analyze**

The analyze process gathers information about the program, including program relationships, logic, data and execution paths, and stores this information in the AKR. After the analyze information is placed in the AKR, it is available to ESW products in online and batch environments, where it is accessed to provide valuable information about the design and operation of user systems. See "[Analyze](#)" on page 339 for more information.

### **analyze options**

Run-time options that control analyze processing. Most are similar to the COBOL compiler options. Default settings are established at installation time and can be overridden by editing the Analyzer JCL or by using the Analyze screens. "[Analyze](#)" on page 339 contains a complete description of each Analyze option.

### **Analyze Summary report**

A summary of the run-time statistics and diagnostic messages that are produced when an Analyze job completes.

### **Application Knowledge Repository**

A BDAM or VSAM file organization that contains all of the analysis information produced by the Analyze job. See ["AKR Utilities" on page 361](#) for more information.

### **Batch AKR command**

Control cards input in the VIASYSIN DD dataset of the Batch AKR Utility. See ["AKR Utilities" on page 361](#) for more information.

### **Batch AKR utility**

Used to maintain the AKR. See ["AKR Utilities" on page 361](#) for more information.

### **COBOL subset**

COBOL verbs of a similar nature that are grouped together. For example, READ, WRITE, OPEN, CLOSE are grouped into the IO subset.

### **command**

Instructions issued by you to Insight. There are three types of commands: primary commands, line commands, and batch AKR utility commands.

### **command input area**

The field on Insight screens where primary commands are entered, indicated by ==> on the fourth line of a screen, or the second line of a pop-up.

### **control flow**

Describes how execution can transfer from one COBOL statement to another.

### **cursor position**

The current location of the cursor on the screen, used as a starting point for certain commands.

### **cursor substitution character**

A token substitution character that may be used in some primary commands. The Cursor character is set on the Options - Product Parameters pop-up. See the online help for more information.

### **data flow**

Describes the path of executable statements leading to how the value of datanames are tested, used, and changed.

### **dataname**

The standard COBOL term for fields defined in the DATA DIVISION of a COBOL program. Variable names, files, groups, array elements, and fully-qualified datanames. Multiple datanames can be searched for at once using plus sign (+). For example: FX ZIP-CODE + HLD-ZIP.

**data usage**

Defines how a data item is used. A data item can be used in one of four ways: 1 USE - when the value is used or tested, 2 MOD - when the value is set or changed (modified), 3 DEF - indicates the statements in the DATA DIVISION where it is defined, 4 REF - means any one of the above.

**DBC**

See ["Double Byte Character Set \(DBCS\)" on page 405](#).

**decision options**

Choices listed on the Trace Decision Menu. Decision options represent the branches in the path that the TRACE command is following.

**decision point**

A statement that causes a branch in the execution path of the program. It could be a PERFORM, conditional statement, exception condition, input statement, GO TO or LABEL.

**Double Byte Character Set (DBCS)**

A character set that uses two bytes to represent each character. Various Double Byte Character Sets are used with languages such as Chinese and Japanese that cannot be represented with single byte codes.

**equate**

A substitution name for a character string. See the ["EQUATE Command" on page 201](#) or the Options - Scratchpad Equate pop-up for more information.

**help**

Insight Help has three levels of help: Long messages, NOTES, and tutorial screens. Specific command information is available by entering the command and pressing PF01/13. The Help facility can also be accessed from the Help pull-down or any Insight screen. See ["Help" on page 171](#) and ["Help Facility" on page 391](#) for more information.

**label name**

Any paragraph or section name of the PROCEDURE DIVISION, as well as the literals PROCEDURE and PROC. Label name specifies all transfers of control to a paragraph or section.

**line commands**

Commands that affect specific lines or blocks of lines, entered in the line command area (columns 1 through 6).

**list file**

A file that is allocated the first time the LPRINT command is executed. This file is used for LPRINT output. See ["LPRINT Command" on page 250](#) or the online help for the Options - Log/List/Punch Definition pop-up for more information.

**live exit**

An abnormality in program control caused by out of perform range GO TO and overlapping perform ranges.

**log file**

The file that is allocated by Insight and used for error messages and log commands. See the online help for the Options - Log/List/Punch Definition pop-up for more information. There is a separate log file created by the Batch AKR Utility. See "[AKR Utility Log](#)" on [page 375](#).

**long message**

A diagnostic or error message that displays on line five of a screen or line three of a pop-up. A long message is sometimes preceded by a short message that displays in the upper right corner of the screen. If the short message does not provide enough information, press PF01/13 to display the long message.

**member**

A member in a PDS or source manager such as Panvalet or Librarian. This can be the alias name found in the AKR.

**network**

The result of a FLOW command. It consists of all possible executable statements that reach from the starting point of the command to all possible targets. The next FLOW command replaces the previous network with the new results.

**online help**

See "[help](#)" on [page 405](#).

**paragraph name**

Any PROCEDURE DIVISION paragraph or section name, and the PROCEDURE and PROC literals. Paragraph name includes the entire paragraph or section.

**path**

Any group of executable statements that describe possible execution flows of the COBOL program. FLOW creates paths called NETWORK and TRACE creates a path called a TRACK.

**perform range**

A perform range consists of the source code contained in a PERFORM statement, and includes all code that is or could be executed as a result of GO TOs, PERFORMs, etc. within that range.

**pop-up**

A window that displays as the result of selecting an item on a pull-down or pop-up, or as the result of entering certain commands. It is superimposed on the screen to allow entry of information for the requested action. See the *ASG-Insight User's Guide* for additional information.

**primary command**

An instruction entered in the command input area of the screen.

**program**

The program source member name, the name specified in the IDENTIFICATION DIVISION of a COBOL program, or the CSECT name of a program that is not COBOL.

**pull-down**

The list that displays when an action is selected on the action bar. On a pull-down, actions followed by ... displays a pop-up when selected. Actions not followed by ... immediately activate internal commands. See the *ASG-Insight User's Guide* for additional information.

**punch file**

A file that is allocated the first time the LPUNCH command is executed. This file is used for LPUNCH output. See ["LPUNCH Command" on page 257](#) the online help for the Options - Log/List/Punch Definition pop-up for more information.

**recursion**

A perform range or paragraph that performs itself.

**SBCS**

[See "Single Byte Character Set" on page 408.](#)

**screen**

A full-width display of information containing an action bar as the first line. See the *ASG-Insight User's Guide* for additional information.

**screen subset**

Lines acted upon by an interactive command that has caused them to be in one of these screen display set types:

- Highlighted | HI
- Excluded | X
- NONHighlighted | NHI
- NONExcluded | NX

**script file**

A dataset containing primary commands entered during a Source View session. The commands are captured automatically when the SET SCRIPT ON or SET SCRIPT RESULT command is issued. See ["SET Command" on page 311](#) for additional information.

**short message**

A diagnostic or error message that displays in the upper right corner of a screen. A short message is sometimes followed by a long message that displays on line five of a screen or line three of a pop-up. If the short message does not provide enough information, press PF01/13 to display the long message.

**Single Byte Character Set**

A character set that uses one byte to represent each character. Single Byte Character Sets are used with languages such as English where the characters can be represented with a one-byte code.

**SMS**

See ["Storage Management Subsystem" on page 408](#)

**Storage Management Subsystem**

An operating environment that helps automate and centralize the management of storage. To manage storage, SMS provides the storage administrator with control over data class, storage class, management class, storage group, and ACS routine definitions.

**subset**

A COBOL subset, screen subset, or tagged lines subset. See the *ASG-Insight User's Guide* for additional information.

**tagged lines subsets**

Information shown in columns 73 through 80 as the result of a FLOW or TRACE command. Tags can be used as subsets in commands that accept subsets as targets.

**target**

The object of an Insight primary command.

**track**

The result of a TRACE command. Track is the set of lines captured while moving through program logic selecting the branch options. See ["TRACE Command" on page 315](#) for more information.

**VIASUB**

An edit macro included with the Insight product that is used to submit an Analyze job.

**VIASUBDS**

A CLIST included with the Insight product that is used to submit an Analyze job.

**work file**

A file that is allocated upon entry into Insight. See the Options - Product Allocations pop-up for more information.

## Symbols

\$\$SEGMENTS program name 19  
& (Retain) command 182, 186

## A

Abend Codes screen 399

action bar 1

AKR

- allocate 15, 20
- allocate without ISPF 377
- analyzed program information 352
- available space in 18
- delete a program from 18
- expand 15, 20
- expand without ISPF 377
- list programs in 17
- management 361
- metrics member 19
- profile member 19
- rename a program in 15, 18
- VIASAKRA JCL example 378
- VIASAKRX JCL example 381

AKR library 8

AKR parameter 346

AKR Program Selection pop-up

- description 7
- field descriptions 8
- SELECT command 310
- VIEW command 323

AKR Utility Log

- description 375
- example 376

alarm 151

Alliance

- accessing from ESW screen xv
- description xii
- linking xv

ALLOCDEF command 187

analysis types 19

analyze

- features 341
- methods 342

- options 341, 355

- parameters 346

- process 342

- requirements 340

- using ISPF 343

- using ISPF/PDF Edit 343

- VIAIN DD statement 349

- VIASUB edit macro 13, 343, 348

- VIASUBDS CLIST 342–343, 348

ANALYZE command 188

analyze process, primary inputs 340

Analyze Submit Parameters screen

- description 346

- example 346

- field descriptions 347

- option descriptions 346

Analyze Submit pop-up 355

Analyze Summary report 352

AOPT parameter 346

AutoChange

- accessing from ESW screen xv

- description xii

Automated Task - Debug a Program Abend

- pop-up

- description 134

- example 134

- field description 134

Automated Task - Modify a Computation

- pop-up

- example 129

- field description 129

Automated Task - Modify Program Output

- pop-up

- example 123

- field description 123

Automated Task - Result Options pop-up

- example 137

- field description 137

automatic JCL modifications 349

**B**

Batch AKR Utility  
  AKR Utility Log 375  
  command format 365  
  command syntax 365  
  comments 367  
  control cards 364  
  CONVERT command 367  
  DELETE command 369  
  EXPORT command 370  
  HELP command 370  
  INIT command 371  
  JCL 364  
  MOVE command 371  
  PRINT command 372  
  print directory example 376  
  print directory report 376  
  PUNCH command 373  
  punch directory 376  
  punch directory example 377  
  Utility Log 375  
BRANCH command  
  description 189  
  on Tree View screen 34  
Branch Previous Options pop-up  
  BRANCH command 190  
  example 95  
  field descriptions 95  
branching logic 190  
Bridge  
  accessing from ESW screen xv  
  description xii

**C**

CALL command 192  
CANCEL command 193  
Center, description xii  
character string 201, 248  
CLIST, VIASUBDS 342–343, 348  
CMPL parameter 346  
COB2R3 analyze option 355  
COBOL intelligent search 216  
COBOLII analyze option 355  
command  
  & (Retain) 182, 186  
  ALLOCDEF 187  
  ANALYZE 188  
  BRANCH 34, 189  
  CALL 192  
  CANCEL 193  
  COPY 194  
  CURRENT 197  
  DELETE 198

END 34, 151, 200  
EQUATE 201  
EXCLUDE 203  
EXECUTE 209  
FIND 212, 293  
FINDXTND 203, 216  
FLOW 228  
GOBACK 151, 233  
HELP 234  
HIGH 235, 294  
JUMP 34, 242  
KEYS 180, 243  
LEVELS 34, 245  
LIST 246  
LOCATE 7, 248  
LPRINT 154, 250  
LPUNCH 154  
MARK 264  
MERGE 267  
PARMDEF 270  
PREF 271  
  primary 210  
PRINTLOG 274  
PRINTLST 275  
PROCESS 276  
  processing 180  
PRODLVL 280  
PROGRAM 151, 281  
QUALIFY 282  
RECALL 180–181, 283  
  recalling 181, 283  
REDO 182, 286  
REFRESH 288  
RENAME 289  
REPEAT 181, 290  
  repeating 181, 290  
RESET 34, 291  
Retain 186  
RETURN 151, 292  
RFIND 182, 293  
RHIGH 182, 294  
RPREF 182, 295  
RSCROLL 182, 296–297  
RTRACE 182, 298  
RTREEVW 301  
SAVE 151, 202, 302  
SCROLL 303  
SELECT 310  
  sequence 180  
SET 311  
SOURCEVIEW 312  
STRUCTURE-VIEW 313  
TRACE 287, 315  
TREEVIEW 321

- VIEW 323
- ZOOMIN 35
- ZOOMOUT 35
- command input area 180, 290
- commands
  - diagram syntax 183
  - line 180, 291, 325
  - on Tree View screen 36
  - primary 180
- comment lines 231
- compiler directives 231
- COND Tree View tag 33
- control, transfer of 383
- conventions page xviii
- COPY command 194
- CUA overview 1
- CURRENT command 197
- cursor position 182, 231
- cursor substitution character 183
- customer support 176
  
- D**
- dataname 201
- DB2LIB analyze option 355
- DB2PLAN analyze option 356
- Debug Abend Task - Line Number Entry
  - pop-up
    - example 135
    - field descriptions 135
- Debug Abend Task - Statement Selection
  - pop-up
    - description 135
    - example 136
    - field descriptions 136
- decision options 287
- decision point 287, 319
- DELETE command 198
- DSCHK parameter 346
- dynamically called programs
  - in analyze job 356
- DYNCALL analyze option 356
  
- E**
- edit macro, VIASUB 343, 348
- EDIT parameter 346
- Encore
  - accessing from ESW screen xv
  - description xiii
- END command 151
  - description 200
  - on Tree View screen 34
- enhancement tasks, automating 121
- EQUATE command 201
  
- equates
  - list 142
  - saving 151, 302
- Estimate
  - accessing from ESW screen xv
  - description xiii
- ESW
  - description xi
  - invoking products xiv
  - product integration xv
- EXCLUDE command 203
- excluded lines 291, 331, 334–335
- EXECUTE command 209
- execution paths 228
  
- F**
- F (First) line command
  - description 327
  - on Tree View screen 36
- features, overview 2
- File - AKR Allocate/Expand pop-up
  - description 20
  - example 21
  - field descriptions 21
  - online utilities 362
  - option descriptions 21
- File - AKR Directory pop-up
  - description 17
  - example 17
  - field descriptions 17
  - online utilities 362
- File - AKR Utility pop-up
  - description 15
  - example 15
  - field descriptions 16
  - online utilities 362
- File - Analyze Submit pop-up
  - description 12
  - example 12
  - field descriptions 13
  - option descriptions 12
- File - Close Program Request pop-up
  - description 10
  - example 10
  - field description 10
- File - Execute Script pop-up
  - example 24
  - field descriptions 24
- File - Open Program Request pop-up
  - description 5
  - example 6

- File - Save Program Request pop-up
  - description 11
  - example 11
  - field description 11
- File - SmartEdit Request pop-up
  - example 23
  - field descriptions 23
- File pull-down
  - actions 4
  - example 4
- FIND command 212, 293
- FINDXTND command 203, 216
- FLAG(x) analyze option 357
- FLAGE analyze option 357
- FLAGW analyze option 357
- FLOW command 228
- G**
- GOBACK command 151, 233
- H**
- H (highlight) line command 241, 328
- help
  - for a command 174
  - for abends 398
  - for commands 395
  - for general information 397
  - for messages 399
  - for pop-ups 394
  - for screens 394
  - for specific information 397
  - how to obtain 393
  - index 397
  - navigational commands 394
  - NOTES 396
  - online 391
  - printing messages 400
  - product release information 175
  - table of contents 397
  - topics 393
- Help - About pop-up
  - description 175
  - example 175
  - field descriptions 176
- Help - Specific ASG Command pop-up
  - description 174
  - example 174
  - field descriptions 174
- Help - Specific ASG Message Number pop-up
  - example 175
  - field descriptions 175
- HELP command 234
- help explanation and action panel 400
- Help pull-down
  - action descriptions 172
  - description 172
  - example 172
- help tutorial 393
- HH (Highlight block) line command
  - description 329
- HH (highlight block) line command 241
- HIGH command 235, 294
- highlighted set, adding lines 241
- highlighting 328–329
- I**
- IN-Clause Option pop-up
  - description 97
  - example 97
  - field descriptions 97
- information tags 100, 319
- INPUT analyze option 357
- INS parameter 346
- Insight
  - accessing from ESW screen xv
  - description xiii
  - facilities 1
  - introduction 1
  - using analysis functions xv
- IO analyze option 357
- J**
- J (Jump) line command
  - description 330
  - on Tree View screen 36
- JCL, automatic modifications 349
- JUMP command
  - description 242
  - on Tree View screen 34
- K**
- KEYS command 180, 243
- L**
- L (Last) line command
  - description 331
  - on Tree View screen 36
- label line command 330
- LANGLVL analyze option 358
- LEARN Mode - Generated Command pop-up
  - description 312
  - displaying 160
  - example 312

- LEVELS command
    - description 245
    - on Tree View screen 34
  - line command
    - block format 326
    - F (First) 36, 327
    - H (Highlight) 241, 328
    - HH (Highlight block) 241, 329
    - J (Jump) 36, 330
    - L (Last) 36, 331
    - Label 330
    - on Tree View screen 36
    - S (Show) 36, 332
    - single format 325
    - SS (Show block) 36, 333
    - X (Exclude) 36, 334
    - XX (Exclude block) 36, 335
    - ZI (zoom in) 36, 324, 336
    - ZO (zoom out) 36, 324, 337
  - LINECNT analyze option 358
  - List - CALL Statements pop-up
    - example 141
    - field descriptions 141
  - List - COBOL Subset Names pop-up
    - description 147
    - example 147
    - field descriptions 147
  - List - Equates pop-up
    - description 142
    - example 142
    - field descriptions 142
  - List - Perform Range Names pop-up
    - description 145
    - example 145
    - field descriptions 145
  - List - Program/Subprogram Names pop-up
    - example 146
    - field descriptions 146
  - List - User Marks pop-up
    - description 143
    - example 144
    - field descriptions 144
  - LIST command 246
  - list file 153, 250, 274–275
  - List Menu pop-up
    - description 247
    - example 247
  - List pull-down
    - description 140
    - example 140
  - LOCATE command 7, 248
  - log file 153, 274–275
  - Logic Order Search - COBOL Subset Name
    - pop-up
      - description 107
      - example 107
      - field descriptions 107
  - Logic Order Search - Data Name pop-up
    - example 103
    - field descriptions 103
  - Logic Order Search - Label Name pop-up
    - description 105
    - example 105
    - field descriptions 105
  - Logic Order Search - Line Number Range
    - pop-up
      - example 116
      - field descriptions 116
  - Logic Order Search - Perform Range Name
    - pop-up
      - example 110
      - field descriptions 110
  - Logic Order Search - Subprogram Name
    - pop-up
      - description 114
      - example 114
      - field descriptions 114
  - Logic Order Search - Trace IN-Clause
    - Option pop-up
      - example 119
      - field descriptions 119
  - Logic Order Search - User Mark Name
    - pop-up
      - description 112
      - example 112
      - field descriptions 112
  - Logic pull-down
    - action descriptions 101
    - description 101
    - example 101
  - logical program unit 33
  - LPRINT command 154, 250
  - LPUNCH command 154
- M**
- MAIN analyze option 358
  - maintenance tasks, automating 121
  - MARK command 264
  - mark name, deleting 198
  - marks
    - creating 164
    - deleting 167
    - list 143
    - renaming 169
    - saving 302
  - MERGE command 267

messages  
  long 291, 400  
  recalling 283  
  short 291, 400

Modify Computation Task - Data Name  
  Entry pop-up  
  description 130  
  example 130  
  field description 130

Modify Computation Task - Data Name  
  Selection pop-up  
  description 131  
  example 131  
  field descriptions 131

Modify Computation Task - Statement  
  Selection pop-up  
  example 133  
  field descriptions 133

Modify Output Task - Data Name Selection  
  pop-up  
  description 127  
  example 128  
  field descriptions 128

Modify Output Task - Output Selection  
  pop-up  
  description 124  
  example 124  
  field descriptions 124

Modify Output Task - Statement Selection  
  pop-up  
  description 126  
  example 126  
  field descriptions 126

## N

NETWORK 100, 228  
NOCMPL parameter 346  
NOCOBOLII analyze option 355  
NODSCHK parameter 346  
NODYNCALL analyze option 356  
NOINPUT analyze option 357  
NOINS parameter 346  
NOIO analyze option 357  
NOOUTPUT analyze option 358  
NOPANEL parameter 346  
NORECUR analyze option 359  
NORETURN analyze option 359  
NOREUS parameter 346  
NORNS parameter 346  
NOSD parameter 346  
NOSDR parameter 346  
NOSDX parameter 346  
NOSEQ analyze option 359  
NOSOURCE analyze option 359

NOST parameter 346  
NOSTX parameter 346  
NOTES in help 396

## O

Options - Allocation Definition pop-up 154  
Options - COPY/Include Libraries pop-up  
  description 138  
  field description 138

Options - Log/List/Punch Definition pop-up  
  displaying 250  
  example 155  
  field descriptions 155  
  option descriptions 155

Options - PF Key Definition pop-up  
  example 158  
  field descriptions 158  
  KEYS command 243

Options - Processing Modes pop-up  
  description 160  
  example 160  
  field descriptions 160

Options - Product Allocations pop-up  
  ALLOCDEF command 187  
  description 152  
  displaying 187  
  example 153  
  field descriptions 153

Options - Product Parameters pop-up  
  example 151  
  field descriptions 151

Options - Scratchpad Copy pop-up  
  description 165  
  example 166  
  field descriptions 166

Options - Scratchpad Delete pop-up  
  description 167  
  example 167  
  field description 167

Options - Scratchpad Equate pop-up  
  example 163  
  field descriptions 163

Options - Scratchpad Mark pop-up  
  description 164  
  example 164  
  field descriptions 164

Options - Scratchpad Merge pop-up  
  field descriptions 168

Options - Scratchpad pop-up  
  description 161  
  example 161  
  field description 161

- Options - Scratchpad Rename pop-up
    - description 169
    - example 170
    - field descriptions 170
  - Options - Script File Allocations pop-up
    - description 157
    - example 157
    - fields 157
  - Options - Set Processing Modes pop-up 311
  - Options pull-down
    - action descriptions 150
    - example 150
  - OUTPUT analyze option 358
  - output names, selecting 124
  - OUTPUT parameter 346
- P**
- PANEL parameter 346
  - PARMDEF command 270
  - perform ranges, listing 145
  - Perform Structure View screen
    - description 40
    - displaying 313
    - STRUCTURE-VIEW command 313
  - PF keys 180, 243
  - PGM EXIT Tree View tag 33
  - PGM parameter 346
  - pop-up
    - AKR Program Selection 7, 310, 323
    - Analyze Submit 355
    - Automated Task - Debug a Program
      - Abend 134
    - Branch Previous Options 93, 190
    - Debug Abend Task - Statement
      - Selection 135
    - definition 1
    - File - AKR Allocate/Expand 20, 362
    - File - AKR Directory 362
    - File - AKR Directory pop-up 17
    - File - AKR Utility 15, 362
    - File - Analyze Submit 12
    - File - Close Program Request 10
    - File - Open Program Request 5
    - File - Save Program Request 11
    - Help - About 175
    - Help - Specific ASG Command 174
    - IN-Clause Option 97
    - LEARN Mode - Generated
      - Command 160, 312
    - List - COBOL Subset Names 147
    - List - Equates 142
    - List - Perform Range Names 145
    - List - Program/Subprogram
      - Names 146
    - List - User Marks 143
    - List Menu 247
    - Logic Order Search - COBOL Subset
      - Name 107
    - Logic Order Search - Label
      - Name 105
    - Logic Order Search - Subprogram
      - Name 114
    - Logic Order Search - User Mark
      - Name 112
    - Modify Computation Task - Data
      - Name Entry 130
    - Modify Computation Task - Data
      - Name Selection 131
    - Modify output Task - Data Name
      - Selection 127
    - Modify Output Task - Output
      - Selection 124
    - Modify Output Task - Statement
      - Selection 126
    - Options - Allocation Definition 154
    - Options - COPY/Include
      - Libraries 138
    - Options - PF Key Definition 243
    - Options - Processing Modes 160
    - Options - Product Allocations 152, 187
    - Options - Scratchpad 161
    - Options - Scratchpad Copy 165
    - Options - Scratchpad Delete 167
    - Options - Scratchpad Mark 164
    - Options - Scratchpad Rename 169
    - Options - Script File Allocations 157
    - Program Summary - Copy
      - Members 51
    - Program Summary - Data 50
    - Program Summary - Files 48
    - Program Summary - Preprocessor
      - Statements 55
    - Script File Allocations 157
    - Search - Any/Unknown Type 91
    - Search - Branch Request 93
    - Search - COBOL Subset Name 78
    - Search - Data Name 66
    - Search - Label Name 70
    - Search - Paragraph Name 72
    - Search - Pattern String 75
    - Search - Perform Range Name 81
    - Search - Program Name 84
    - Search - User Mark Name 86
    - Selection List 96, 118
    - Trace Decision Options 298–299, 310
    - View - Exclude Request 59
    - View - Levels Request 57

- View - Paragraph Cross Reference 45
- View - Paragraph Cross-Reference Request 42
- View - Program Summary 47
- View - Structure View Request 37
- View - Tree View Request 30
- PREF command 271
- prefix area 325
- printing messages 400
- PRINTLOG command 274
- PRINTLST command 275
- PROCEDURE DIVISION label 189
- PROCESS command 276
- PROONLY parameter 346
- PRODLVL command 280
- product integration xv
- product level 280
- program
  - list 146
  - structure 324
  - VIASMPRT 400
- PROGRAM analyze option 358
- PROGRAM command 151, 281
- Program Summary - Control Flow pop-up
  - example 54
  - field descriptions 54
- Program Summary - Copy Members pop-up
  - description 51
  - example 51
  - field descriptions 52
- Program Summary - Data pop-up
  - example 50
  - field descriptions 50
- Program Summary - Files pop-up
  - example 48
  - field descriptions 48
- Program Summary - Preprocessor Statements pop-up
  - description 55
  - example 56
- Program Summary - Sections/Paragraphs pop-up
  - example 53
  - field descriptions 53
- program view 298
- pull-down
  - File 4
  - Help 172
  - List 140
  - Logic 101
  - Search 65
  - Task 122
  - View 27
- punch file 153, 257, 274-275

**Q**

- QUALIFY command 282

**R**

- RECALL command 180-181, 283
- Recap
  - accessing from ESW screen xv
  - description xiii
- RECUR analyze option 359
- RECUR Tree View tag 33
- REDO command 182, 286
- REFRESH command 288
- release level 280
- RENAME command 289
- REP Tree View tag 34
- REPEAT command 181, 290
- REPEATED Tree View tag 34
- requesting help 393
- RESET command
  - description 291
  - on Tree View screen 34
- Retain (&) command 186
- RETURN analyze option 359
- RETURN command 151, 292
- REUS parameter 346
- RFIND command 182, 293
- RHIGH command 182, 294
- RNS parameter 346
- RPREF command 182, 295
- RSCROLL command 182, 296-297
- RTRACE command 182, 298
- RTREEVW command 301

**S**

- S (Show) line command
  - description 332
  - on Tree View screen 36
- SAVE command 151, 202, 302
- save equates 151
- screen
  - Abend Codes 399
  - Analyze Submit Parameters 346
  - help explanation and action panel 400
  - Perform Structure View 40, 313
  - Tree View 32, 301
- script file comments 211
- SCROLL command 303
- SD parameter 346
- SDR parameter 346
- SDX parameter 346

- Search - Any/Unknown Type pop-up
    - description 91
    - example 91
    - field descriptions 91
  - Search - Branch Request pop-up
    - description 93
    - example 93
    - field descriptions 93
  - Search - COBOL Subset Name pop-up
    - description 78
    - example 78
    - field descriptions 79
  - Search - Data Name pop-up
    - description 66
    - example 67
    - field descriptions 67
  - Search - Label Name pop-up
    - description 70
    - example 70
    - field descriptions 70
  - Search - Line Range pop-up
    - example 88
    - field descriptions 89
  - Search - Paragraph Name pop-up
    - description 72
    - field descriptions 73
  - Search - Pattern String pop-up
    - description 75
    - example 75
    - field descriptions 75
  - Search - Perform Range Name pop-up
    - description 81
    - field descriptions 82
  - Search - Program Name pop-up
    - description 84
    - example 84
    - field descriptions 84
  - Search - User Mark Name pop-up
    - description 86
    - example 86
    - field descriptions 87
  - Search pull-down
    - action descriptions 65
    - description 65
  - SELECT command 310
  - Selection List pop-up
    - description 96, 118
    - example 118
  - SEQ analyze option 359
  - SET command 311
  - sets
    - copy 143
    - delete 143
    - merge 143
    - rename 143
  - SmartDoc
    - accessing from ESW screen xv
    - description xiii
  - SmartEdit
    - accessing from ESW screen xv
    - description xiv
    - enter from Task 122
  - SmartTest
    - accessing from ESW screen xv
    - description xiv
  - SOURCE analyze option 359
  - Source Manager, specifying 138
  - SOURCE Tree View tag 34
  - SOURCEVIEW command
    - description 312
  - SPACE analyze option 359
  - SQLID analyze option 359
  - SRC Tree View tag
    - description 34
  - SS (Show block) line command
    - description 333
    - on Tree View screen 36
  - ST parameter 346
  - stopping point 319
  - STRUCTURE-VIEW command
    - description 313
  - STX parameter 346
  - SUBNET 228
  - subprogram, internal 146
  - subsets, list 147
  - SUBSYS analyze option 360
  - support, customer 176
- T**
- tags
    - information 100
    - Tree View generated 33
  - Task pull-down
    - action descriptions 122
    - description 122
    - example 122
  - To 146
  - TRACE command 287, 315
  - Trace Decision Options pop-up
    - description 299
    - example 299
    - field descriptions 299
    - RTRACE command 298
    - SELECT command 310
  - TRACK 319
  - transfer of control 383
  - Tree View commands 34

Tree View generated tags 33  
Tree View screen  
  command 301  
  description 32  
  example 33  
  primary commands 34  
  tags 33  
Tree View tag 33  
TREEVIEW command 321

## V

VIAIN DD statement 349  
VIASAKRA JCL 377  
VIASAKRX JCL 377  
VIASMPRT program 400  
VIASUB edit macro 13, 343, 346, 348  
VIASUBDS CLIST 342–343, 346, 348  
View - Exclude Request pop-up  
  description 59  
  field descriptions 60  
View - Levels Request pop-up  
  description 57  
  field description 57  
View - Paragraph Cross Reference  
  pop-up 295  
  description 45  
  example 46  
  field descriptions 46  
View - Paragraph Cross-Reference Request  
  pop-up  
  description 42  
  example 42  
  field descriptions 42  
View - Program Summary pop-up  
  description 47  
  example 47  
  field description 47  
View - Reset Request pop-up  
  example 58  
  field descriptions 58  
View - Structure View Request 37  
View - Structure View Request pop-up  
  field descriptions 38  
View - Tree View Request pop-up  
  description 30  
  field descriptions 31  
VIEW command 323  
View pull-down  
  action descriptions 27  
  description 27  
  example 27

## X

X (exclude) line command  
  description 334  
  on Tree View screen 36  
XLIVE analyze option 360  
XMEM analyze option 360  
XX (exclude block) line command  
  description 335  
  on Tree View screen 36

## Z

ZI (zoom in) line command  
  description 336  
  on Tree View screen 36  
  syntax 324  
ZO (zoom out) line command  
  description 337  
  on Tree View screen 36  
  syntax 324  
ZOOMIN command, on Tree View  
  screen 35  
ZOOMOUT command, on Tree View  
  screen 35



ASG Worldwide Headquarters Naples Florida USA | [asg.com](http://asg.com)