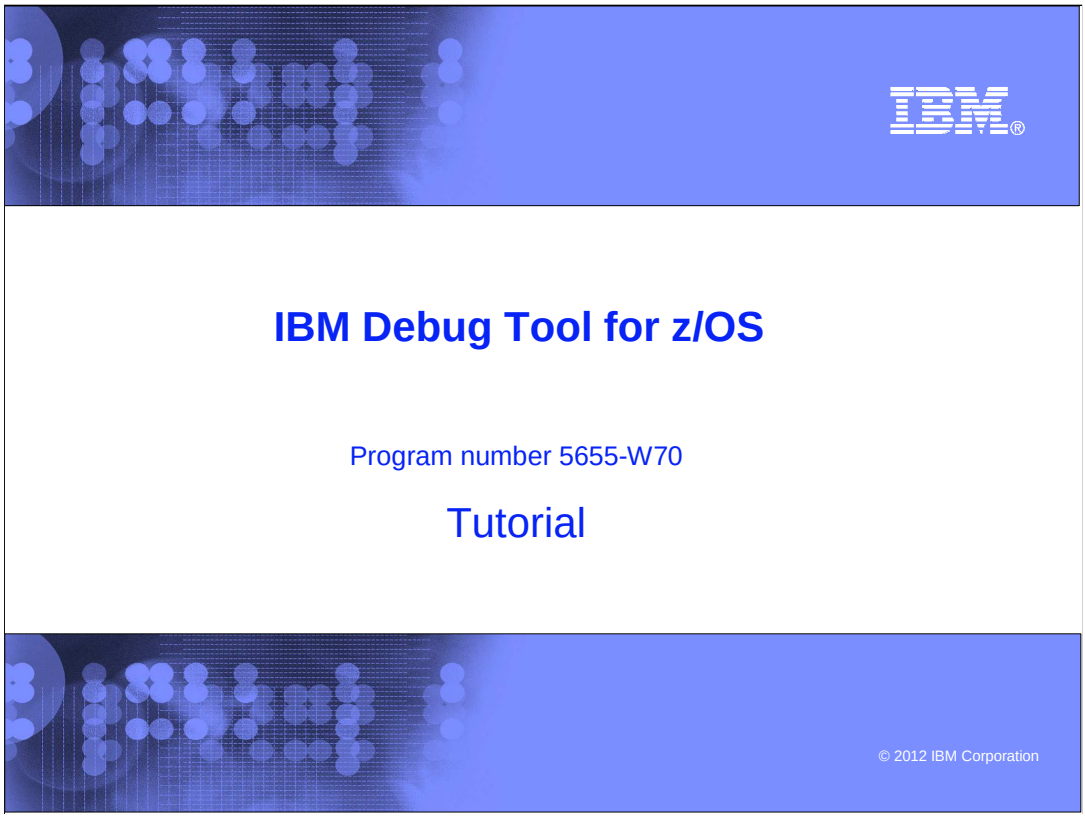
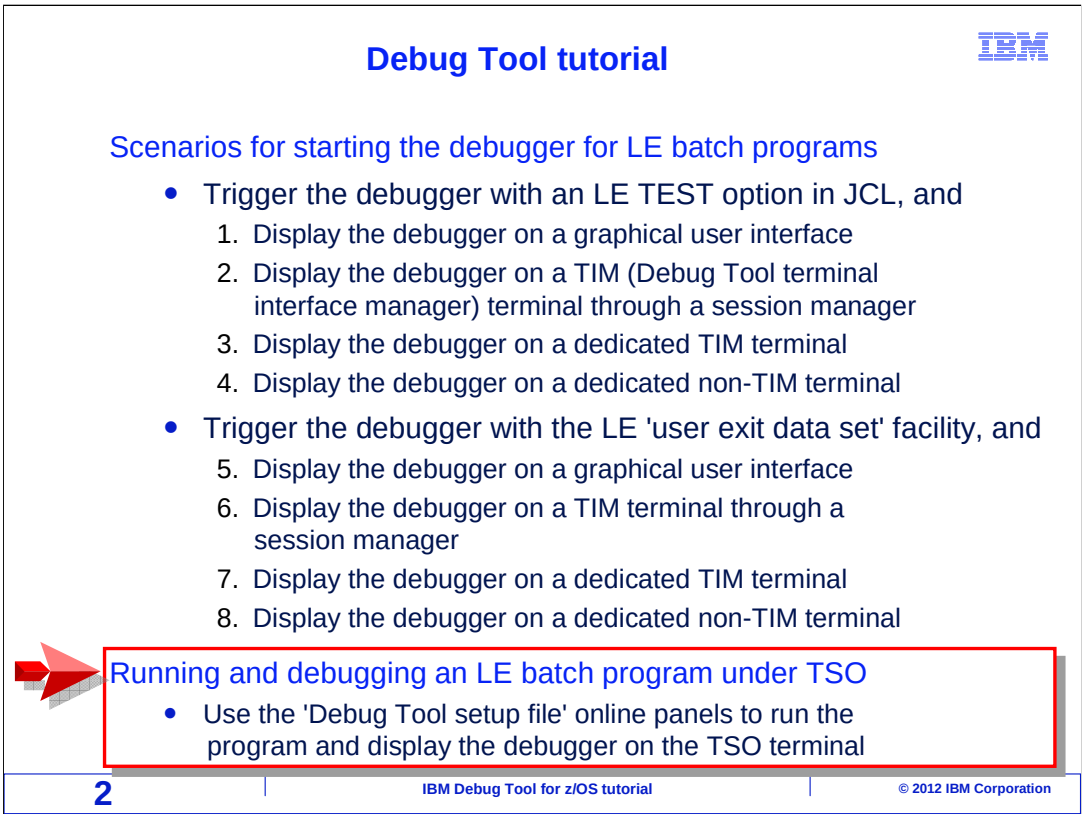


This is the tutorial for IBM Debug Tool for z/OS®, one of the IBM zSeries® problem determination tools.



In this section, you will see a way to run and debug a batch application under TSO, using the 'Debug Tool setup file' online panels. You can skip this section if you do not plan to use Debug Tool this way on your system.

Debug in TSO using an LE TEST option

- Description
  - The Debug Tool Setup file online panels are used to create a 'setup file'
    - A setup file performs a function similar to JCL
    - It names a program to run and has DD statements
  - When you run a setup file, the application runs in TSO
  - An LE TEST option can be used to set a trigger for the debugger
  - The debugger displays on the TSO terminal
- Consider using this method:
  - When batch initiators are not available to run batch jobs on your system
  - If dedicated Debug Tool terminals, or GUI debugging software, are not available on your system
- When not to use this method:
  - If the batch-oriented methods are available. They may be quicker to set up.
  - If you need to debug a program and use TSO services at the same time

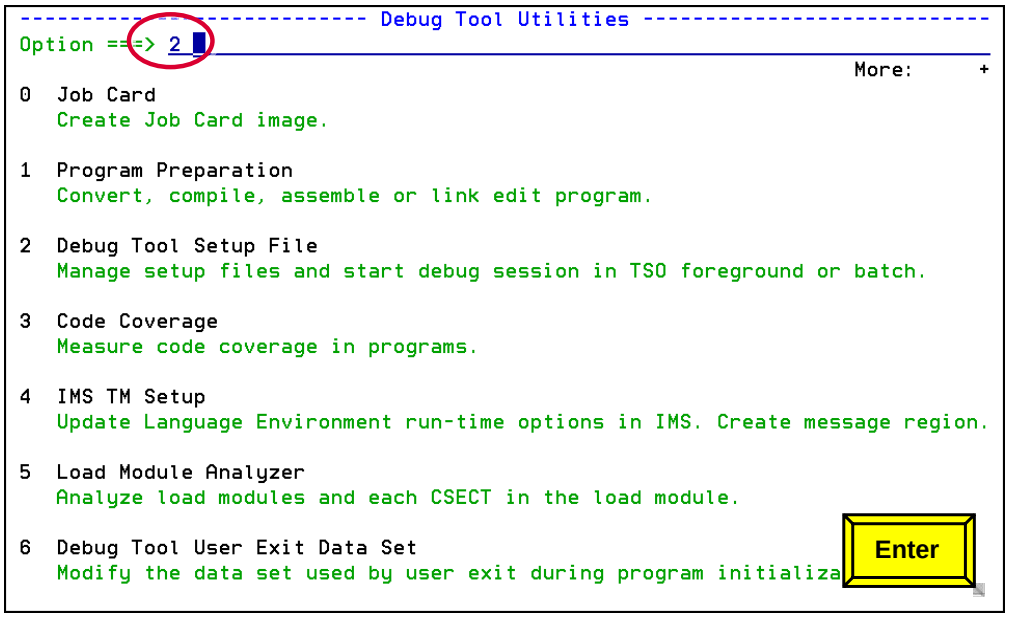
This is an optional way to debug batch programs. A set of online panels in TSO are used, called the 'Debug Tool setup file' utility. The panels are used to create something called a 'setup file', which serves a similar purpose as JCL. When you think of it, JCL is used to specify the name of a load module to run, provide parameters for the program, and specify a list of files that the program will use in the form of DD statements. A setup file has similar information. After you create a setup file with the right information, you run it. The program runs in your TSO session, and has access to the files in the DD list.

If you set a debugging trap in the setup file, the debugger displays on the user's TSO terminal, or on GUI debugging software on your workstation. As with programs running in batch, an LE TEST option can be used to start the debugger.

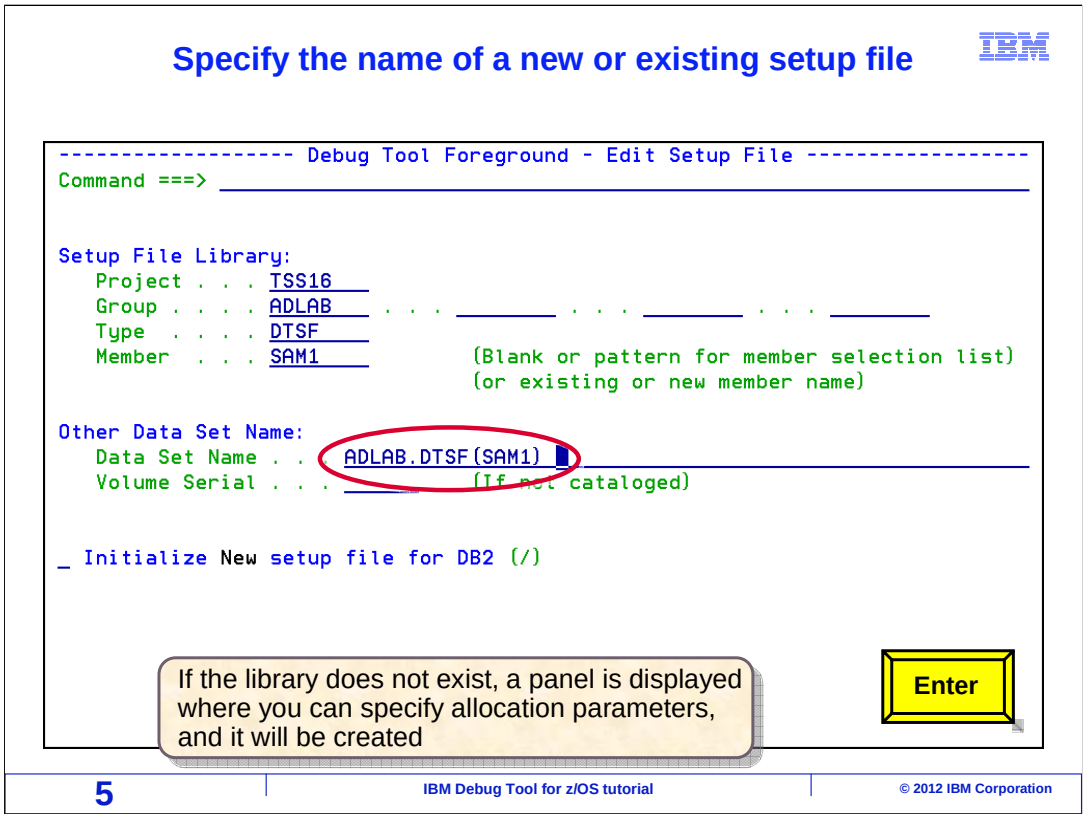
Consider using this method if batch initiators are not available on your system to run batch jobs, or if dedicated Debug Tool terminals or GUI debugging software are not installed on your system.

Typically, this method would not be used if one of the batch-oriented methods is available. The batch methods may be quicker to set up. Also, you may prefer to use one of the batch methods if you need to debug a program and use TSO services (such as the editor or ISPF utilities) at the same time.

Navigate to the Debug Tool Utilities menu
Select option 2 for Debug Tool Setup File

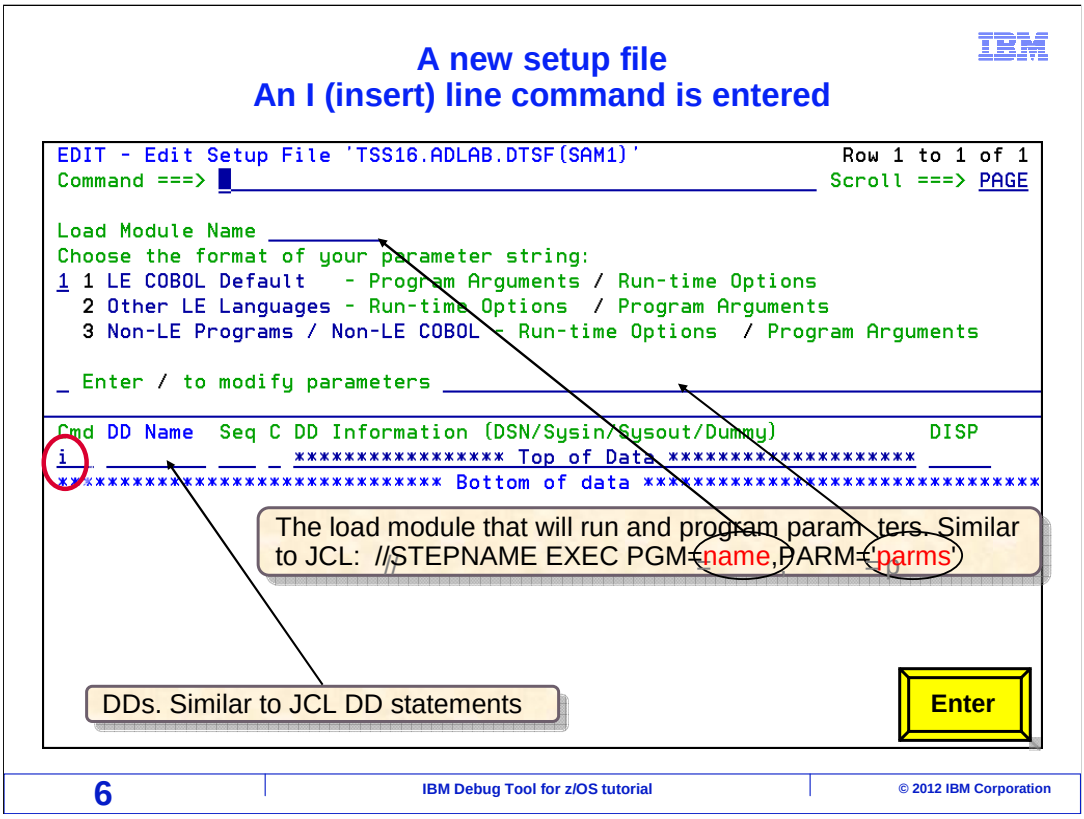


First, navigate to the Debug Tool Utilities panel in TSO. Typically, you can get there using the menus on your system or with a command. Type 2 on the command line to select the 'Debug Tool setup file' option, and press Enter.



You are prompted for the name of a setup file. If you specify a file name, or a PDS member name that does not exist yet, then you are creating a new setup file. If you have setup files, you can specify the name here to use it again. If you specify the name of a file or PDS or library that does not exist, you will be prompted automatically to allocate the new data set.

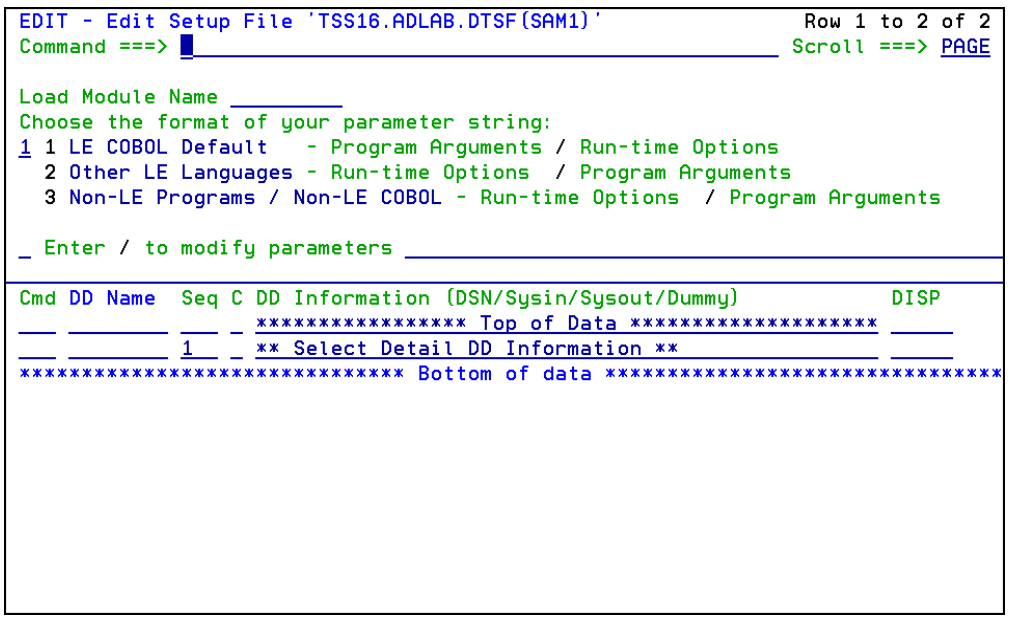
In this example, a PDS already exists where setup files can be saved. The name of the PDS is entered, but with a new member name, so this setup file will be stored as a new member. Enter is pressed.



A new setup file is shown. Notice the three main parts of a setup file: load module name, parameters, and DDs. The load module name can be entered the field near the top. This is the module that will run when the setup file runs. The 'load module name' field serves the same function of an EXEC statement in JCL. Parameters are optional. If your program requires parameters when it runs, type them into the field to the right of the text 'enter / to modify parameters'. Parameters are the same as the PARM option of an EXEC statement in JCL.

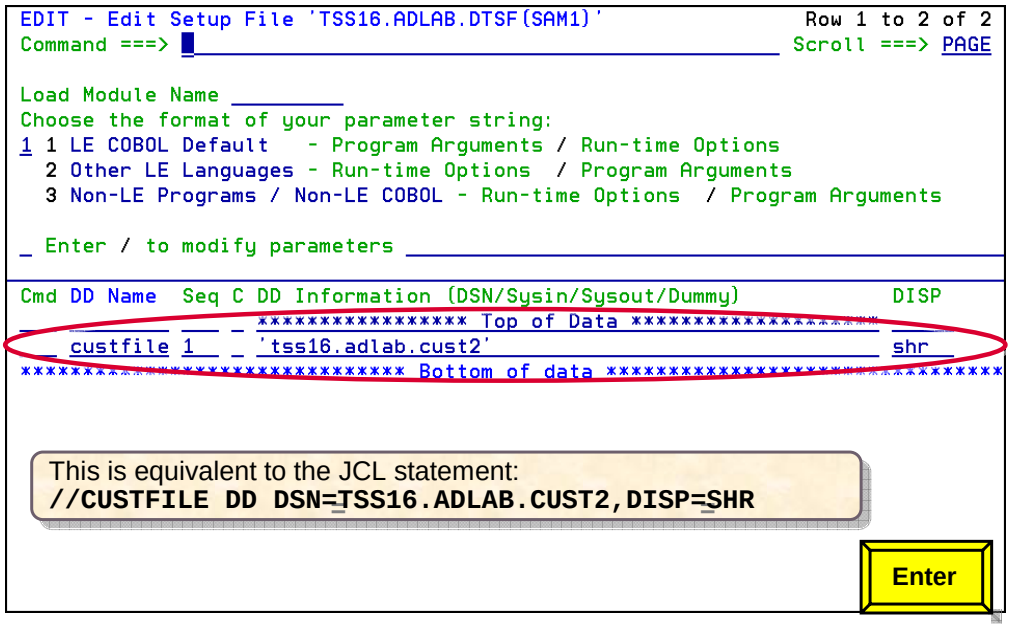
A new setup file does not have any DDs, and they can be entered into the list. To insert a new DD, type an "I" line command in the 'Cmd' field as in this example, and press Enter.

Result of I (insert) line command: a new line for a DD was added



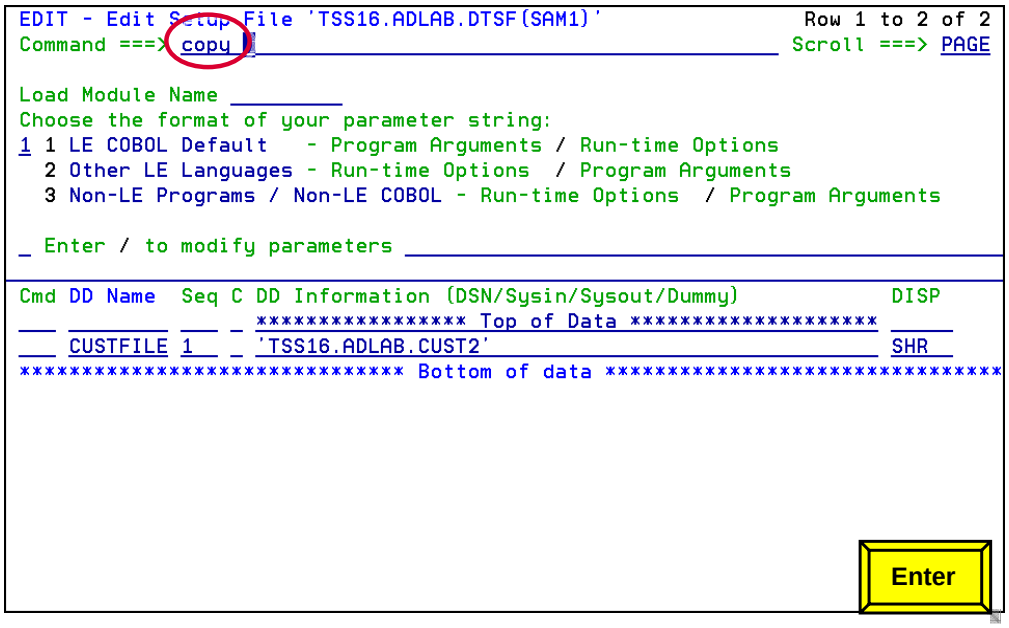
The "I" line command added a new line for a DD.

DDs can be entered this way



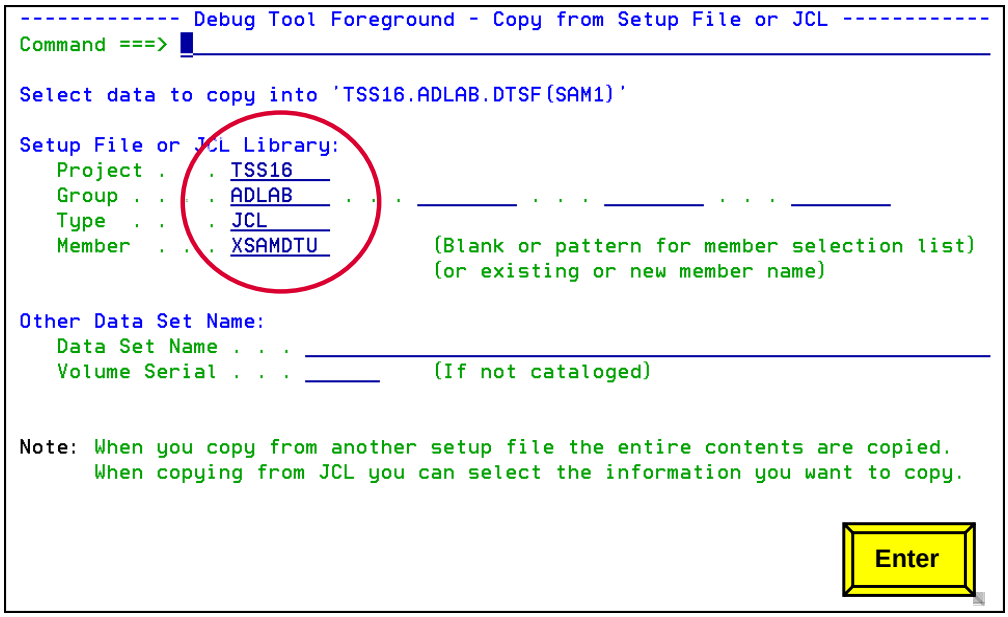
Here, a new DD for CUSTFILE is entered, referencing file 'TSS16.ADLAB.CUST2' with DISP=SHR. It serves the same function as a DD statement in JCL.

Use a COPY command to copy from JCL or another setup file



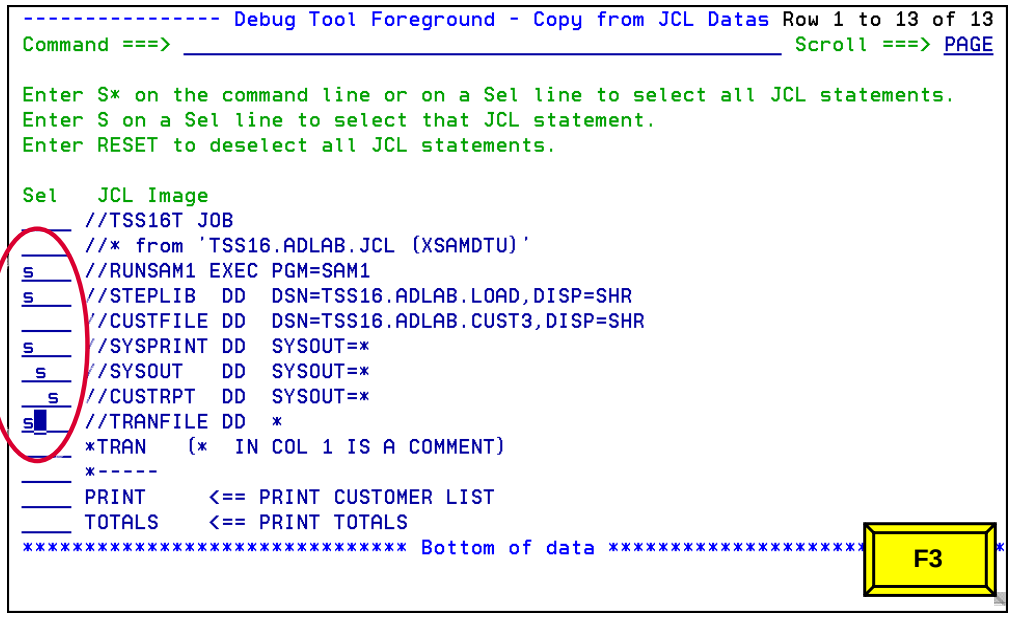
You can continue to insert DD statements manually, one at a time, until all the DDs that your program needs are specified. However, if JCL already exists with the information needed to run the program, there is an easier way. You can copy information from the JCL into the setup file. To do it, type COPY on the command line, and press enter.

You can copy from JCL or another setup file



The COPY command displays the 'Copy from setup file or JCL' panel. From here you specify the name of an existing setup file or JCL that you want to copy. In this example, the name of a PDS member containing JCL is specified, and Enter is pressed.

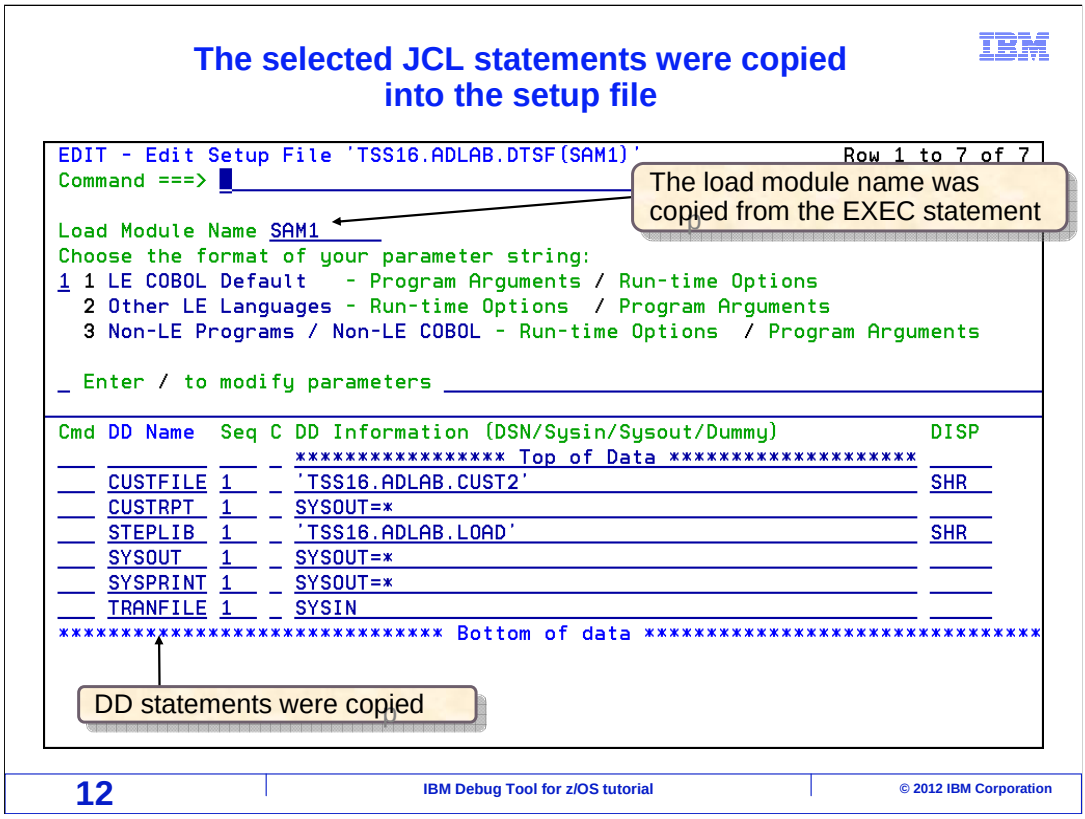
When copying from JCL, select the statements to be copied



The JCL was read, and a list of JCL statements are displayed. Use S line commands to select each of the JCL statements that you want to copy into the setup file.

Notice that the EXEC statement is selected. The name of the load module on the EXEC statement will be copied. If there are parameters on the EXEC statement, they will be copied also.

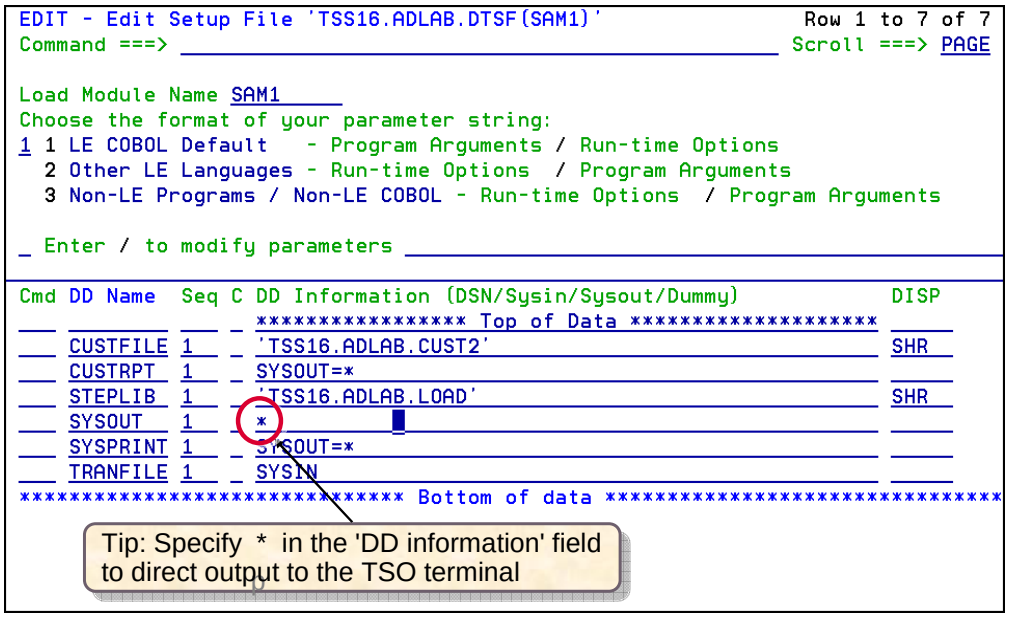
If you want to copy all of the statements in the JCL, you can use an S* command to select everything. After making the selections, press F3.



The selected JCL statements were copied into the setup file. Notice that the load module name, SAM1, was copied along with several DDs. The DD that had been entered manually is still there.

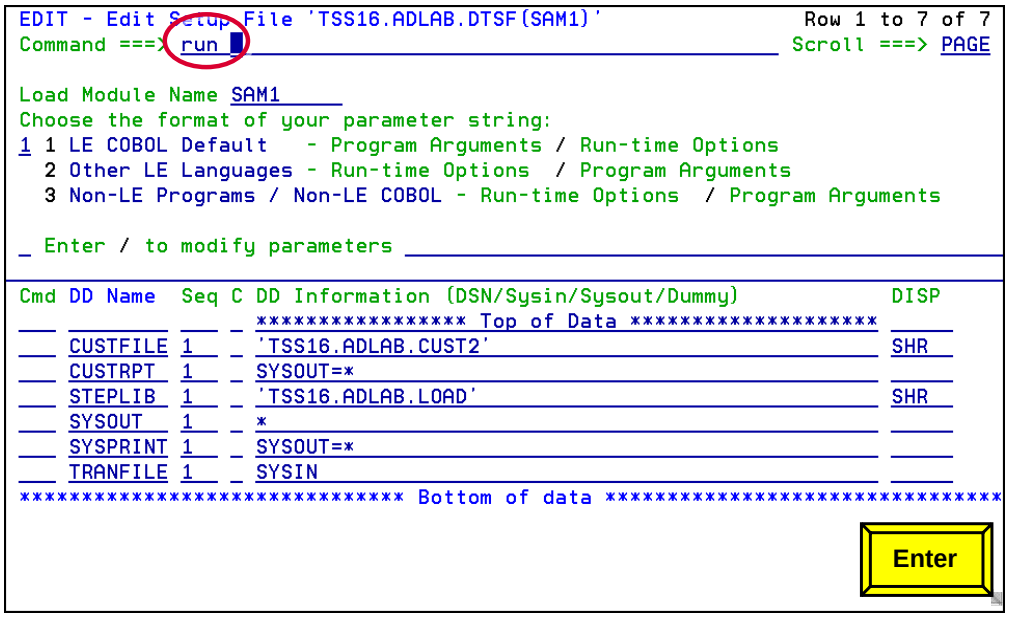
Notice the SYSOUT DD statement. In the JCL, it refers to JES output. If you have DD statements directed to SYSOUT, the output will still go to SYSOUT later when you run the setup file in TSO.

Overtype DD information to change it



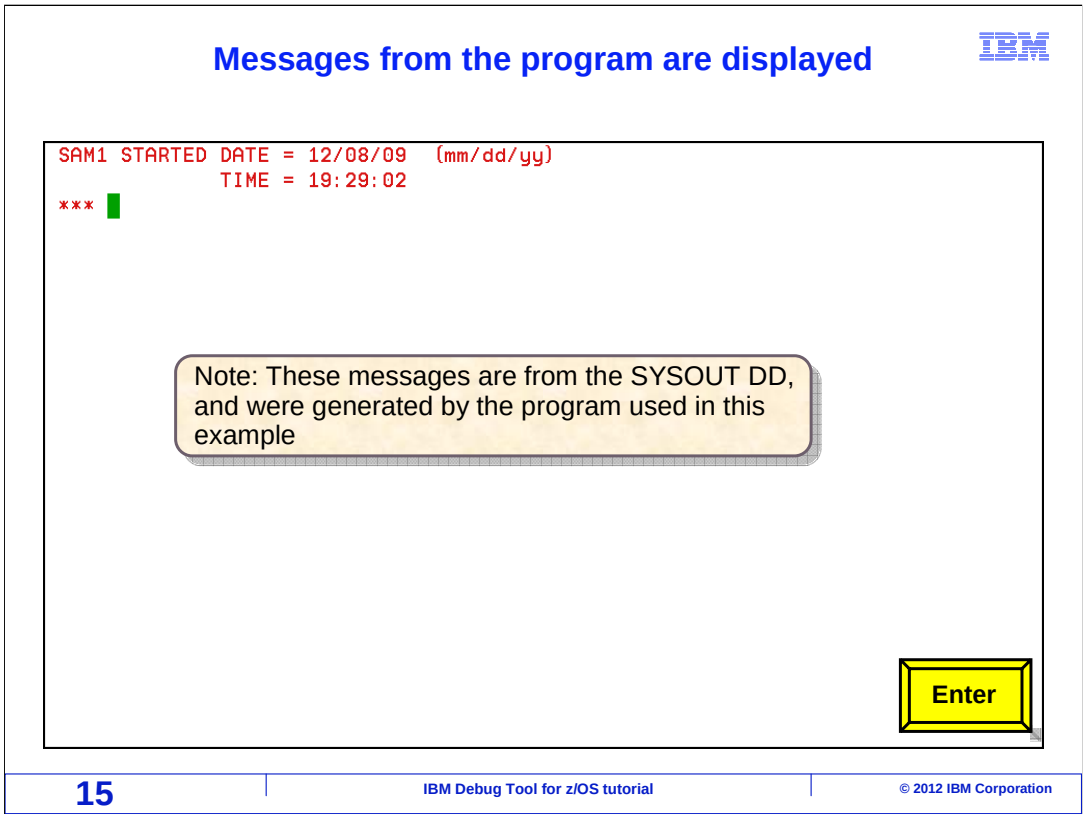
In this example, the DD information for the SYSOUT DD is changed to asterisk (*). This output will be sent to the TSO terminal as the program runs. This can be a good option in some cases if you want to see messages produced by the program automatically on your terminal. However, it is best to avoid this option if the DD will receive a large amount of output.

The RUN command runs the setup file



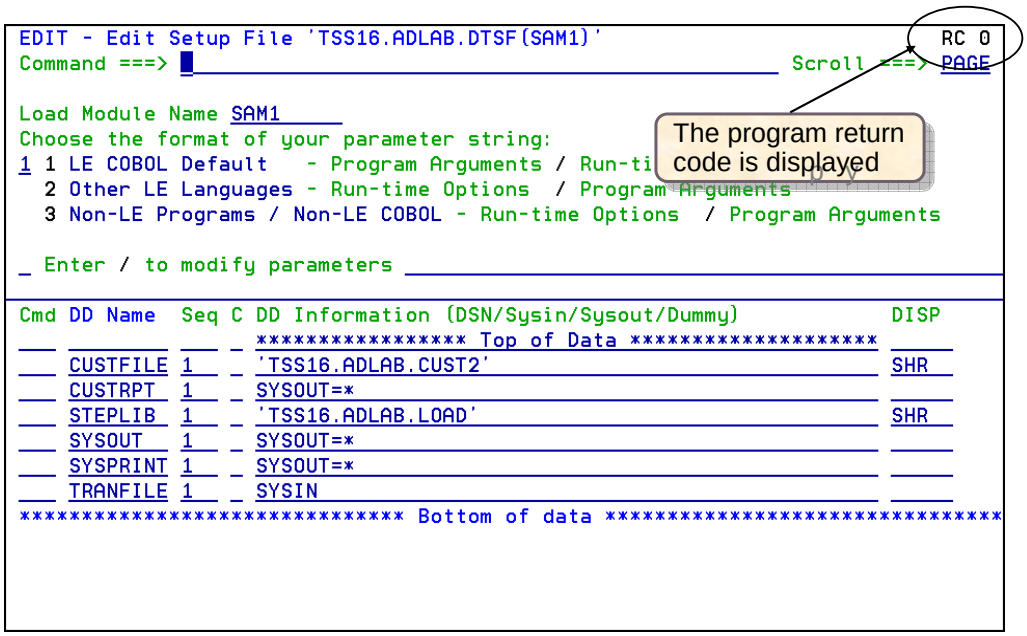
At this point, the setup file is ready to run. A load module name, and all the DDs it needs are specified. This particular program does not need any run-time parameters, so none were entered.

The RUN command runs the setup file.



In this example, the program is writing messages to a DD named SYSOUT as it runs. Remember that the SYSOUT DD was specified so that these messages would be displayed on the TSO terminal. The messages shown here are produced by the program, and are unique to this example. Press Enter to continue.

The program ran



The program ran all the way through to it's termination. Notice that the program's return code is shown.

Triggering the debugger

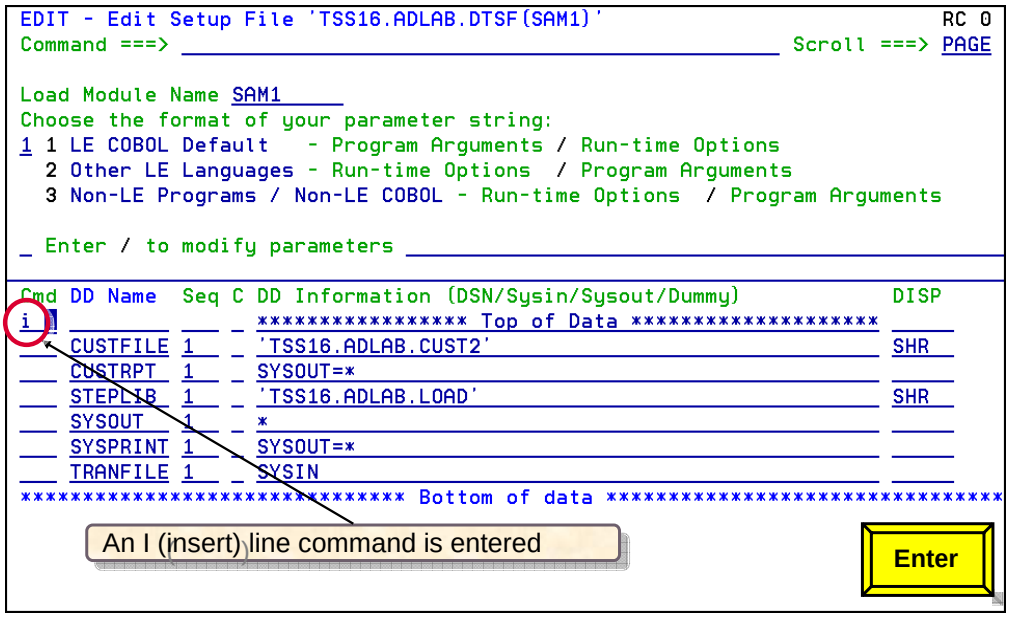
- In this example, the program ran, but the debugger did not start
  - A debugging trigger was not set
- You can use the same methods to start the debugger that are available for debugging batch jobs:
  - A CEEOPTS DD with an LE TEST option
  - The Debug Tool user exit data set facility
  - An LE TEST option coded in the parameter string
    - Tip: Type a slash (/) in the field marked 'Enter / to modify parameters' to display a dialog that will generate a TEST option automatically
- In this example, a CEEOPTS DD will be used

So far, you have seen how to create a setup file to run a batch program in TSO. Running a setup file is a way to run a program, similar to running a program in batch. However, you may have noticed that the debugger did not start when the program ran.

Just as when a program runs in batch, a trigger for the debugger must be set so the debugger will start when the program runs. And the same methods that are available to trigger the debugger for a program running in batch are available when using a setup file. The most common methods include coding a CEEOPTS DD statement with an LE TEST option, using the Debug Tool 'user exit data set' facility, and coding an LE TEST option in the program parameter string.

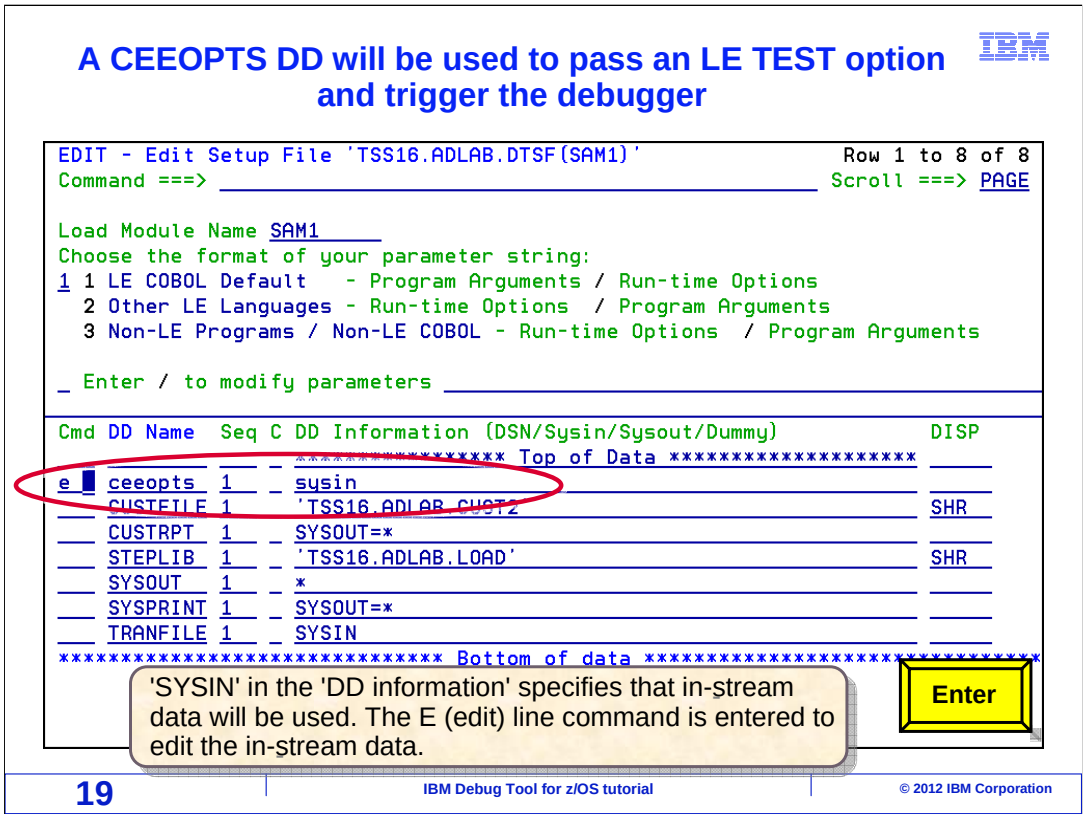
In this example, a CEEOPTS DD statement will be used.

Insert a new DD statement



An I line command is entered to insert a new DD statement.

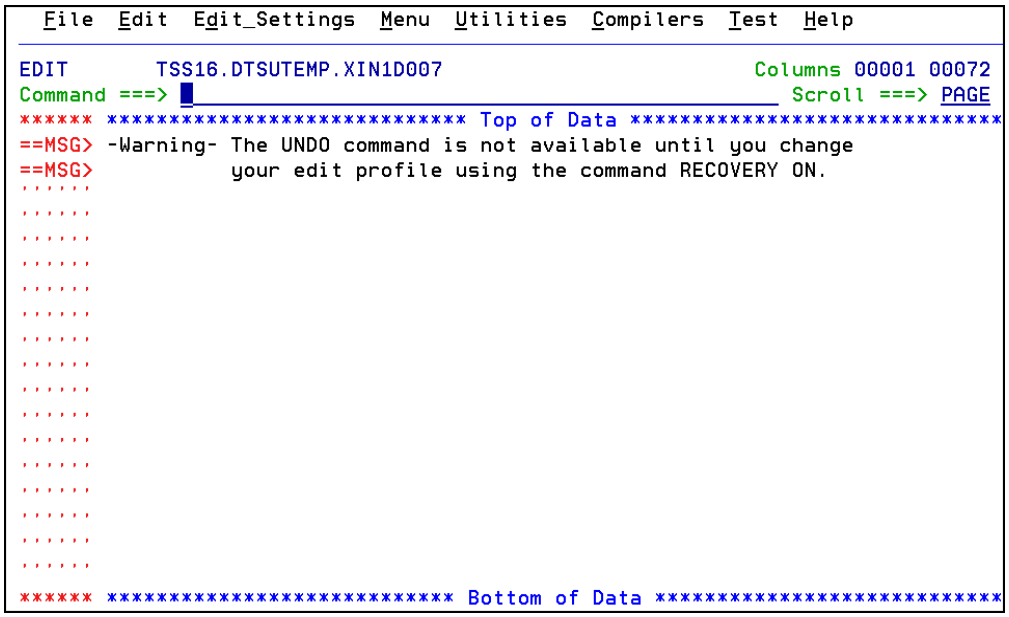
A CEEOPTS DD will be used to pass an LE TEST option and trigger the debugger



The new line is overtyped to specify a CEEOPTS DD. 'SYSIN' is specified in the 'DD information' field. That specifies that in-stream data will be used.

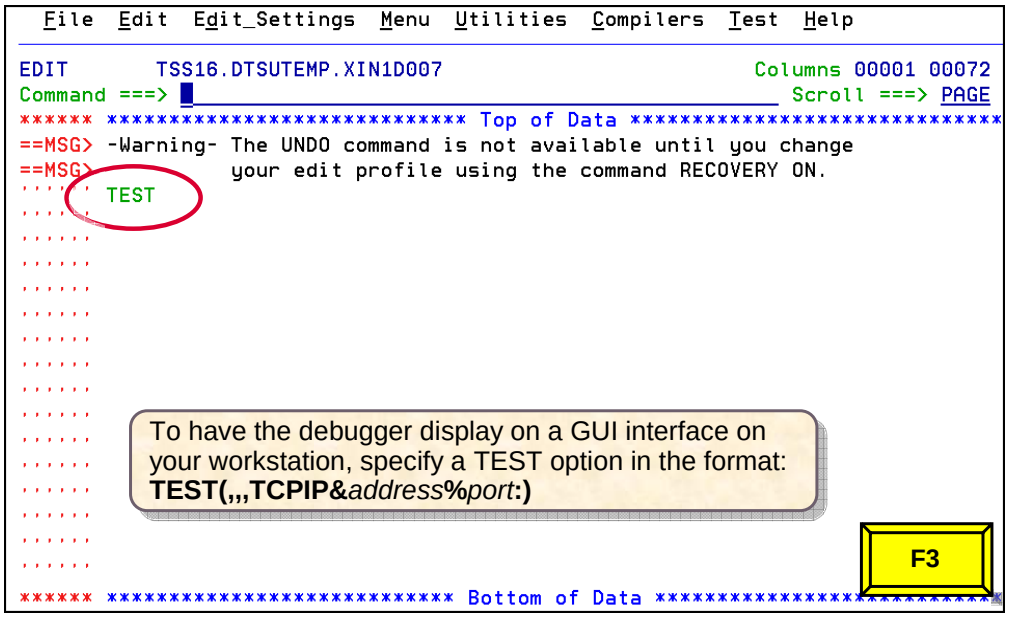
An E (for edit) line command is entered for the CEEOPTS DD. This will display the in-stream data in the editor.

The editor is displayed



The E line command displayed the editor. Since this is a new DD statement, no data has been entered yet.

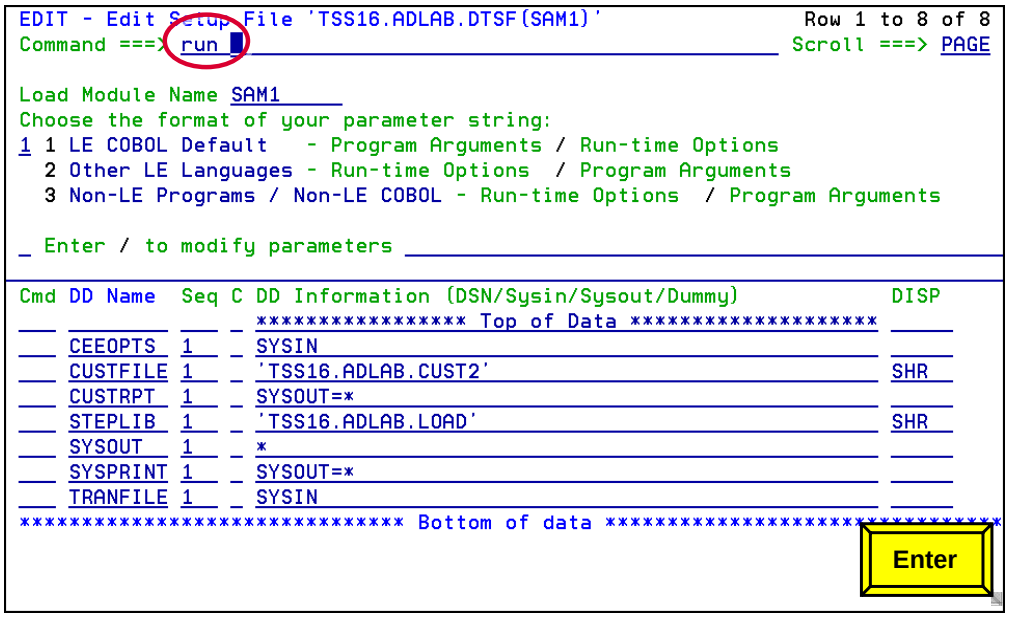
A TEST option is entered



A TEST option is entered. In a setup file, a simple TEST option with no sub-options is all that is needed if you want the debugger to display on your TSO terminal. The LE TEST option has several sub-options you can specify if they are needed or helpful. For example, if you want to have the debugger display on a GUI on your workstation, specify the TCPIP sub-option with your workstation's IP address.

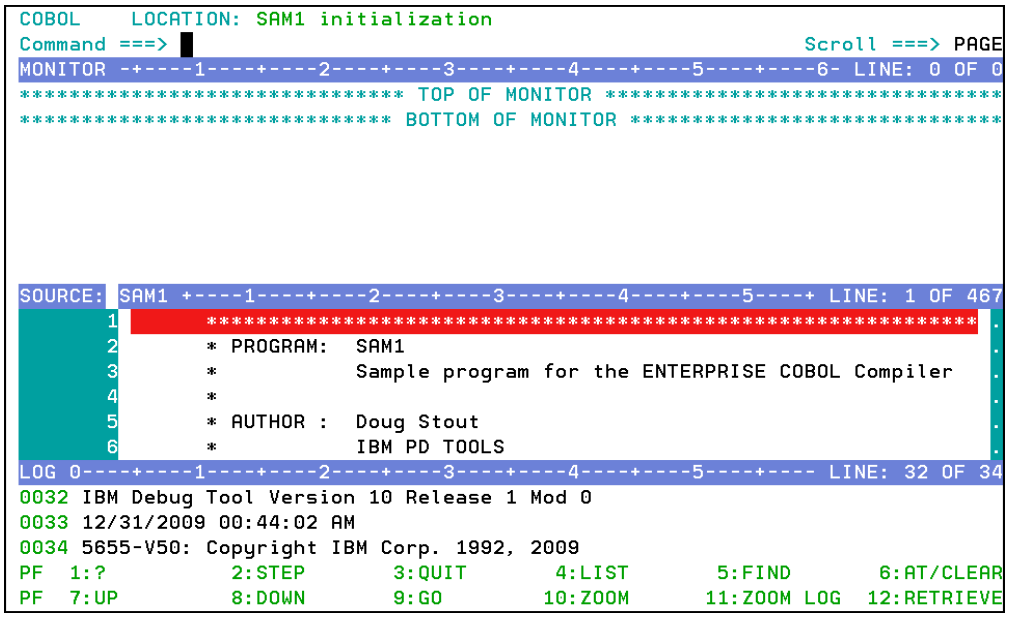
Press F3 to save the data.

The RUN command runs the setup file



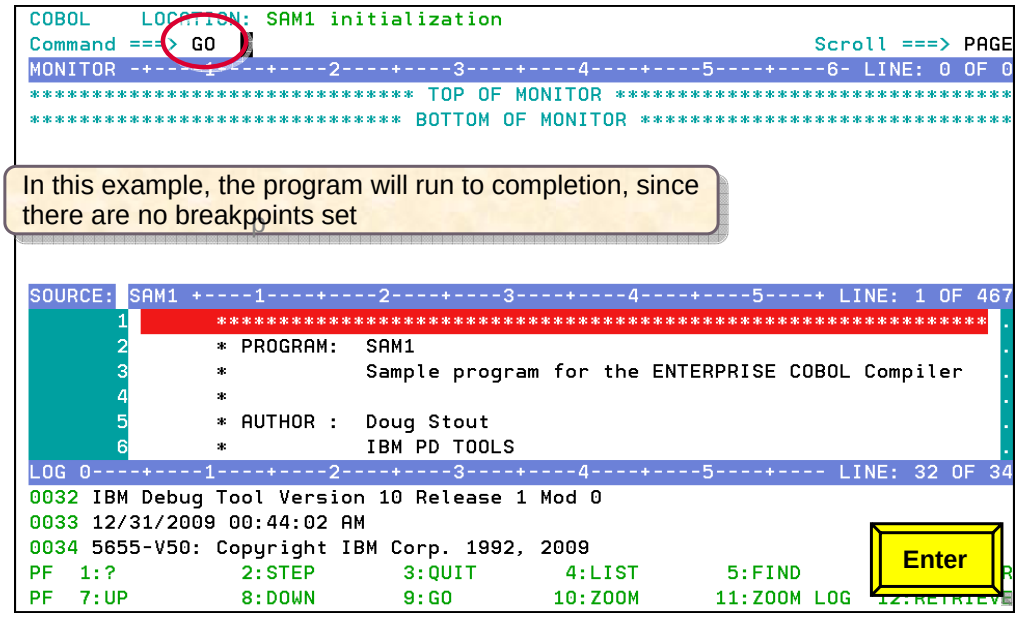
Since the CEEOPTS DD contains SYSIN data, the actual data will be saved within the setup file itself. A RUN command is entered to run the setup file.

The debugger is displayed on the TSO terminal



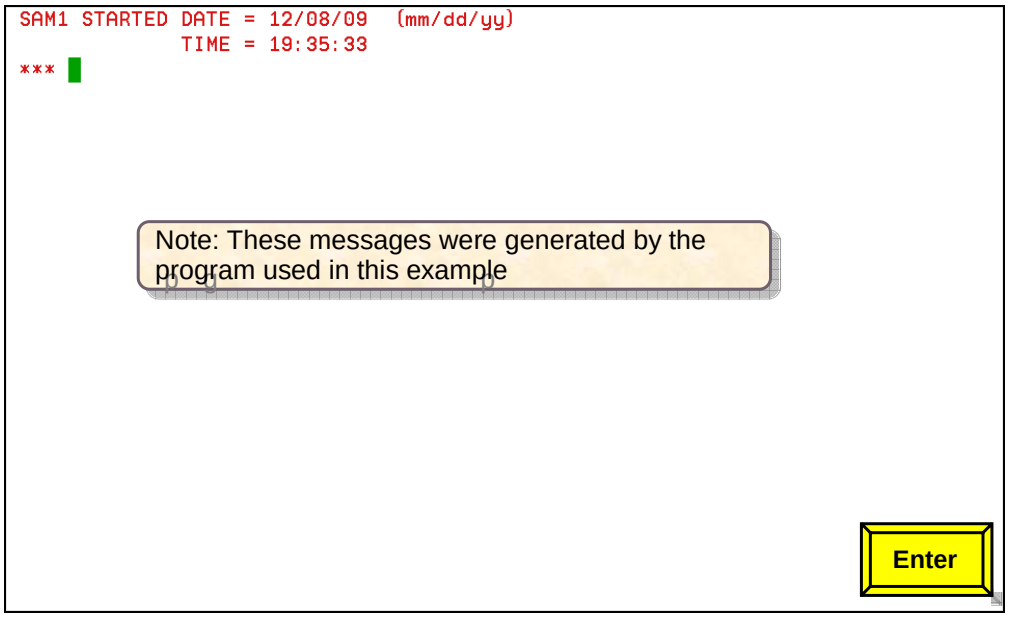
When the first LE program runs, Language Environment checks to see if a CEEOPTS DD statement is present. If it is, it is read for options. The TEST option is a trigger that directs LE to start the debugger. The debugger is displayed on your TSO terminal.

A GO command runs the program

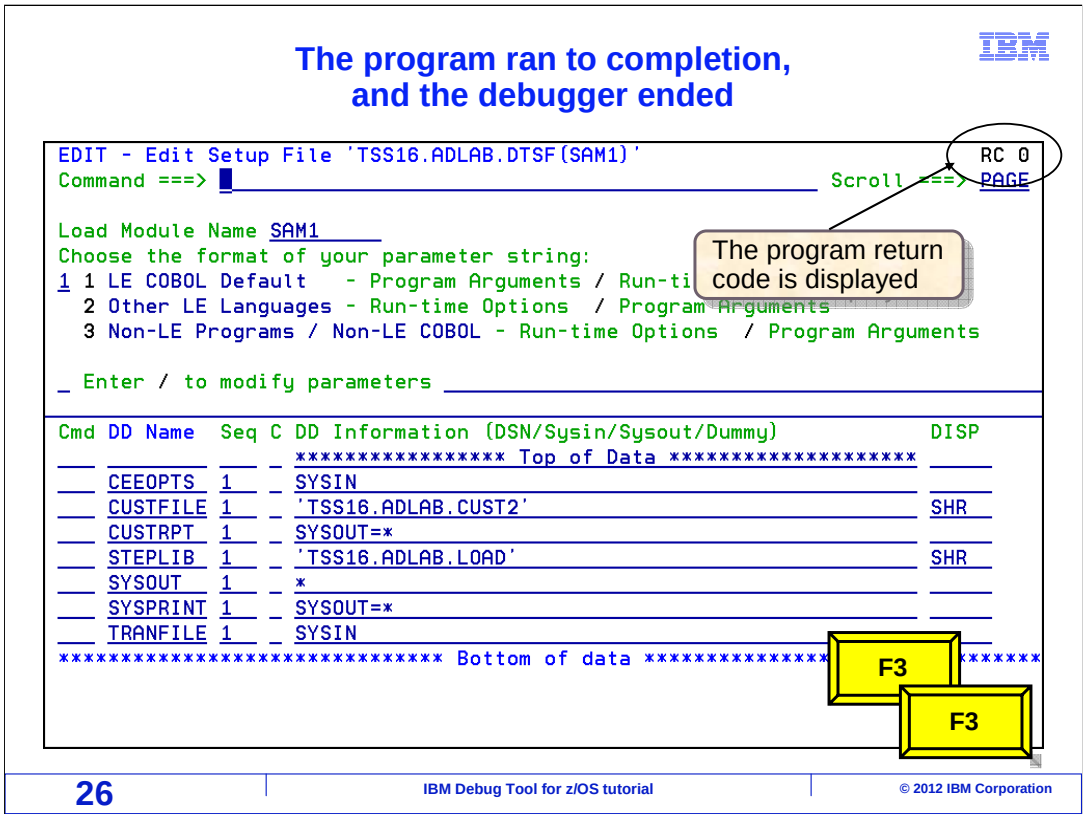


In this example, no breakpoints are set in the program, and a GO command is entered to run the program.

Messages from the program are displayed



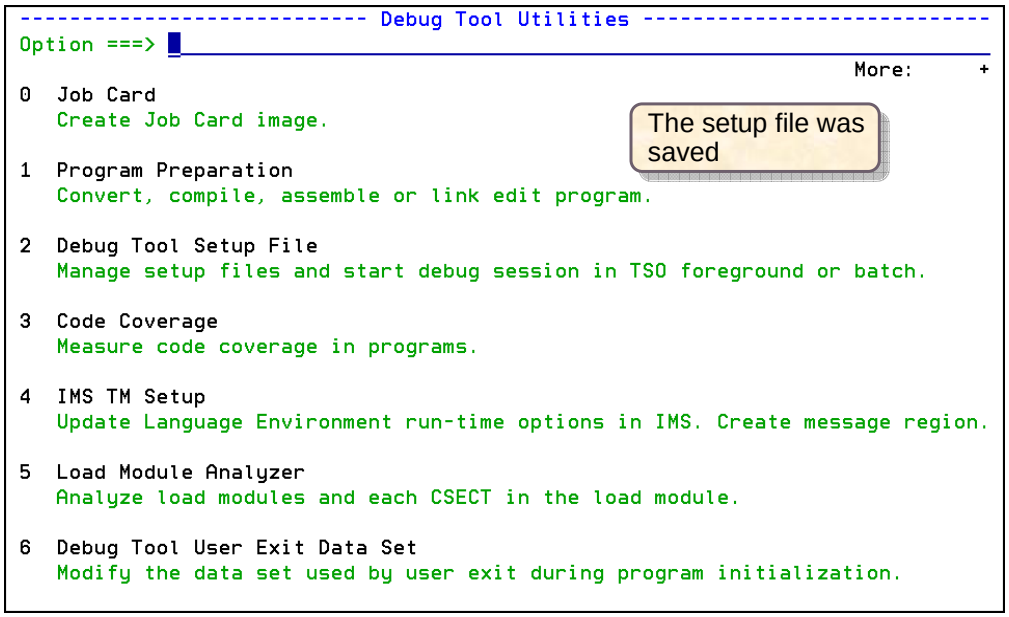
Messages from the application program may be displayed on the TSO terminal. Press Enter to continue.



The program ran to completion. When the program terminated, so did the debugger. The 'edit setup file' panel is displayed again, with the return code from the program.

F3 is pressed twice to return to the Debug Tool utilities menu.

Back to the Debug Tool Utilities menu



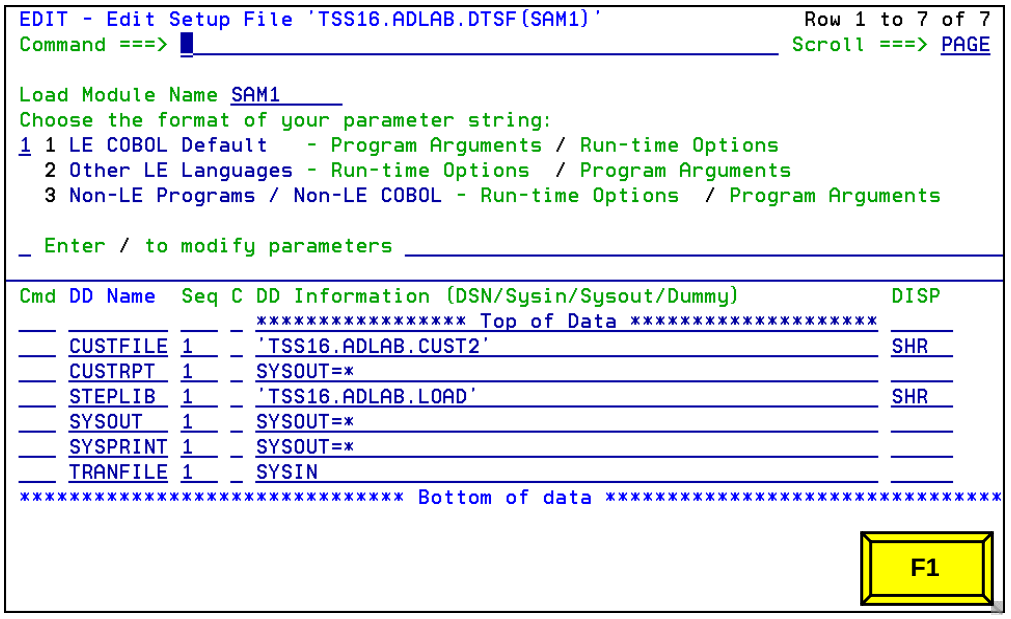
The setup file was saved automatically. It can be re-used at any time to run and debug the program.

More about the setup file panels

- Next, you will see how to:
  - Display a list of available commands
  - Display a list of available prefix commands (line commands)
  - Specify detailed DD information
  - Create a concatenated DD statement (to reference multiple files)

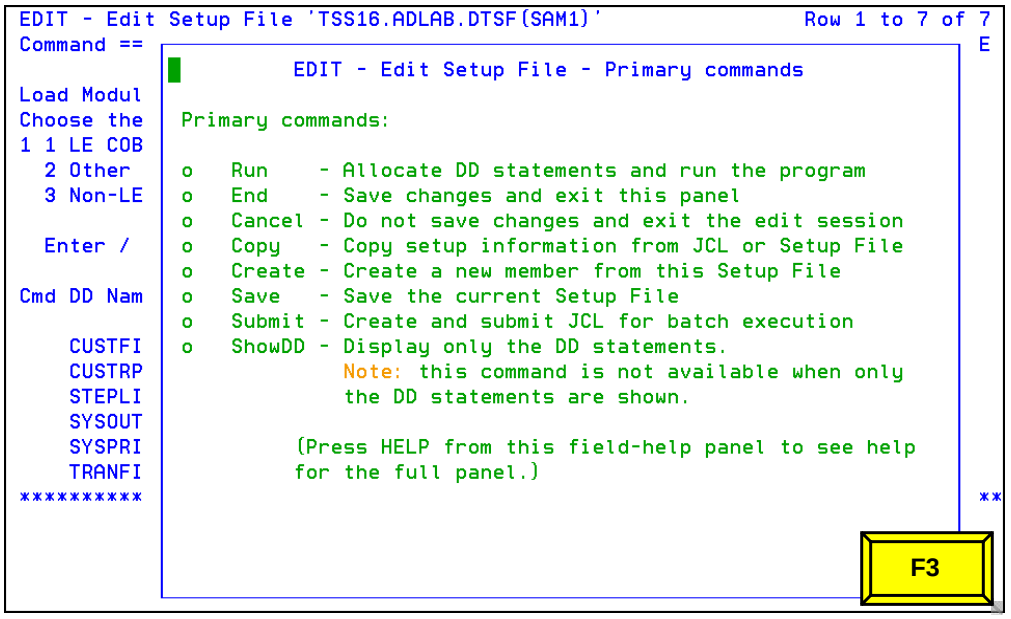
Next, you will see how to display a list of available commands and line commands, specify detailed DD information, and how to specify concatenated DD statements.

Press F1 (help) to see a list of commands



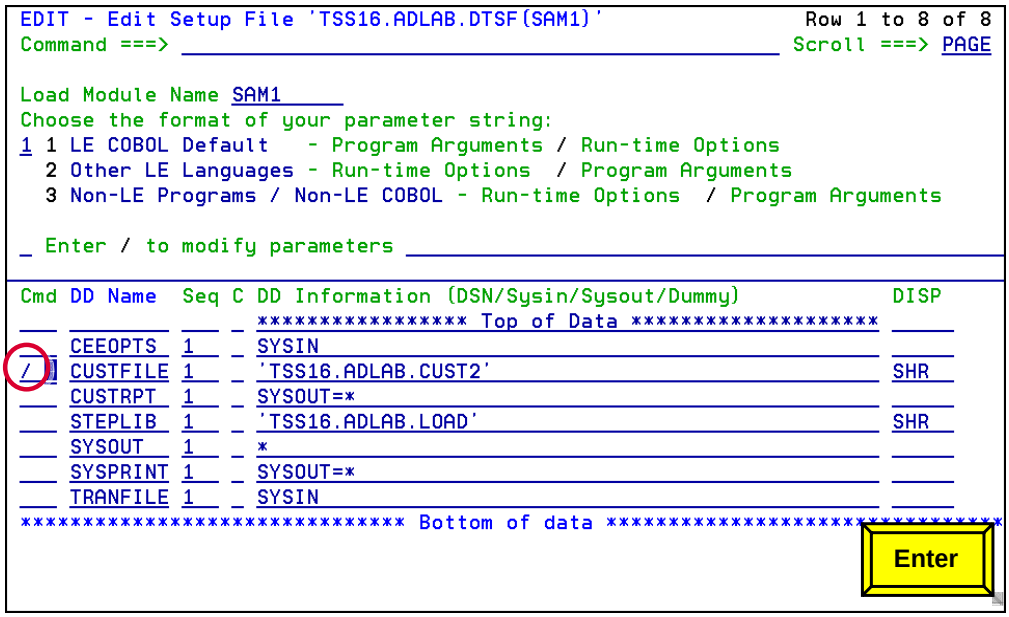
Pressing F1 from the 'edit setup file' panel will display help.

A list of commands available on the setup file panel

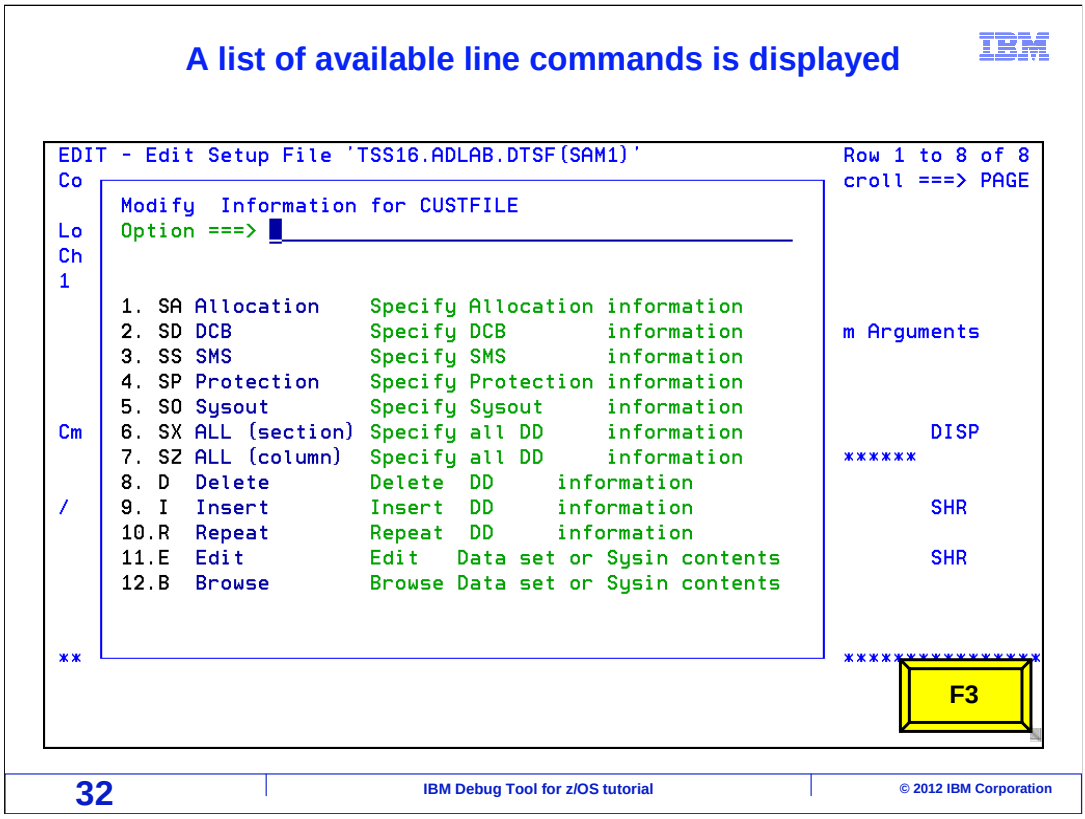


The help panel displays a list of available commands. You have already seen the RUN command used. It runs the setup file. COPY is used to copy information from another setup file that already exists or from JCL. Be aware that the SUBMIT command is available to automatically generate JCL from the setup file. The SUBMIT command lets you actually run JCL in batch. F3 returns from the help panel.

Use a slash (/) line command to see a list of line commands



A slash (/) line command displays a list of available line commands.

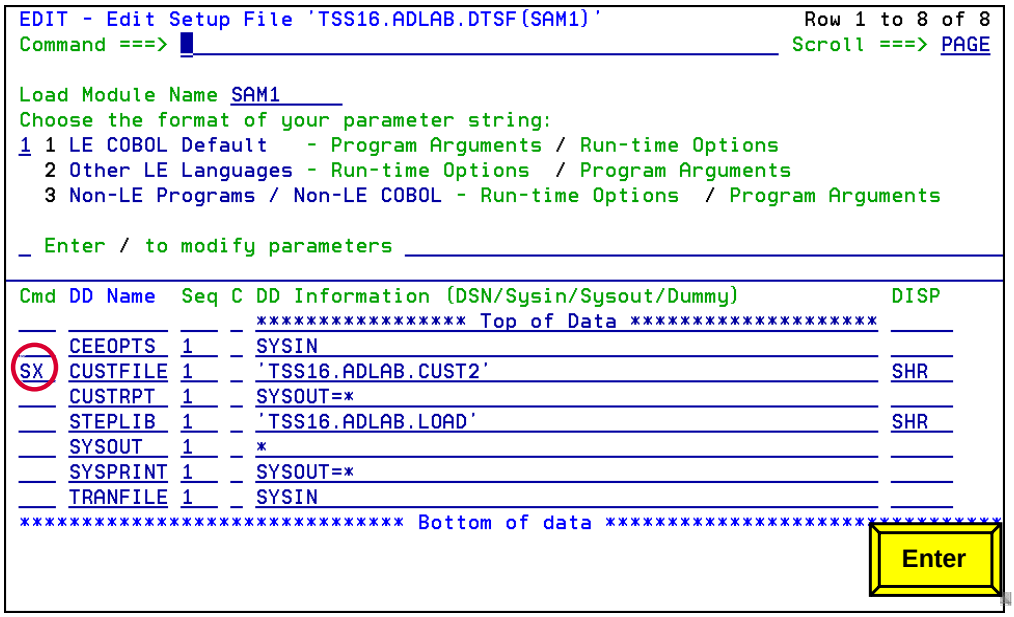


Here are the line commands you can use. There are several line commands that start with the letter S, such as SA and SD. These S line commands provide ways to specify more complex DD information than can be entered on the main panel.

Use a D line command to delete a DD statement, I to insert a new one, and R to repeat DD information to a new line to build a concatenation. If a DD refers to a sequential file, a PDS member, or in-stream data, you can use an E line command to edit the data in the file. A B line command is used to browse data in a file.

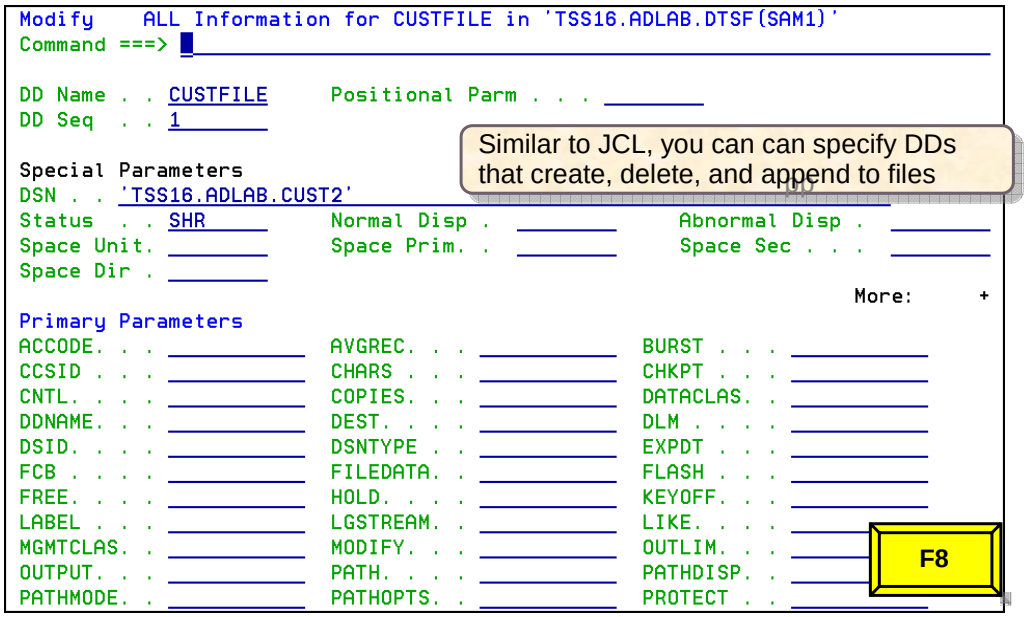
Press F3 to exit the line command help panel.

The SX line command displays all available DD information



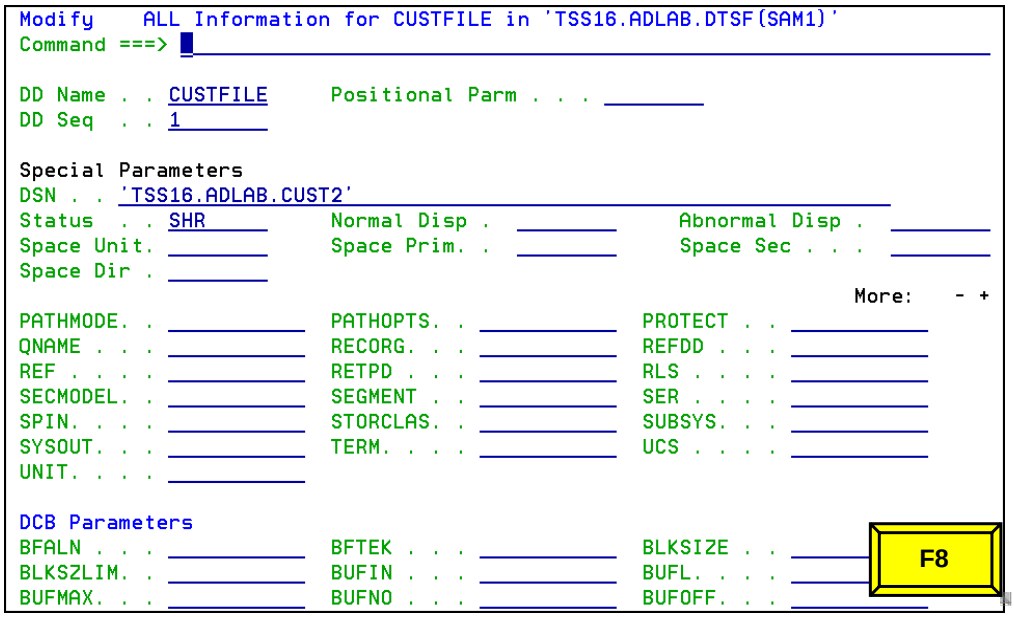
On the 'edit setup file' panel, you can see the list of DDs. But only basic DD information, such as file name and disposition, can be entered. To specify more detailed DD information, use one of the line commands that begins with S. For example, here, an SX line command is entered on the CUSTFILE DD.

Result of SX (select all DD information) line command



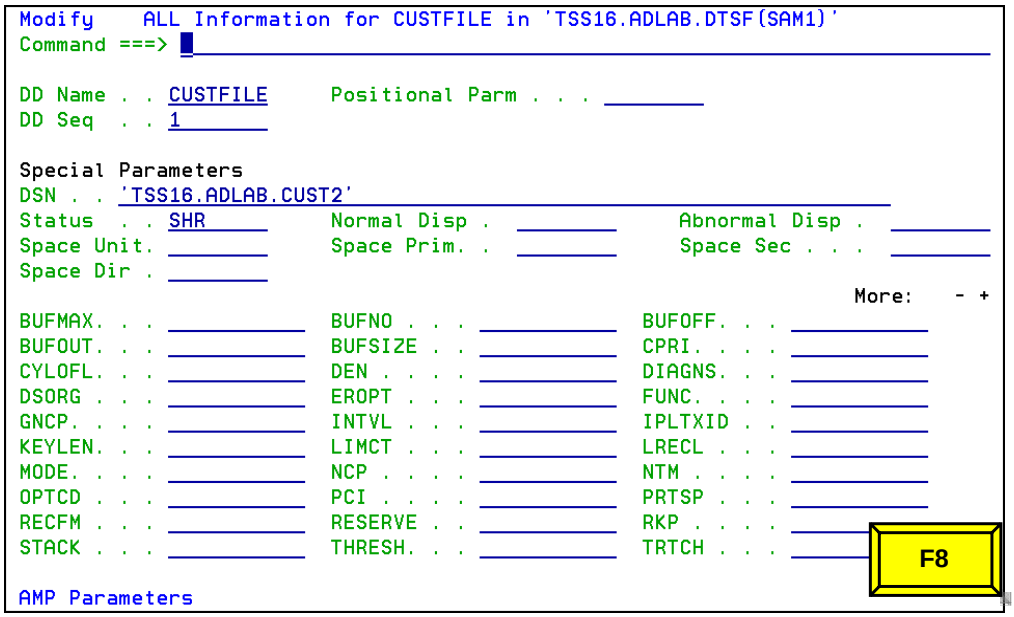
The SX line command displays the 'All information' panel. From here, you can specify more complex DD information, similar to what can be done in JCL. For example, if you intend to have a new file created when the setup file runs, you can specify NEW in the 'Status' field, and specify allocation information including entries in the 'Space unit', 'Space prim'(ary), and other fields. There are so many options that they do no all fit on one panel. F8 is pressed to display more options.

Additional options available from the SX (select all DD information) line command



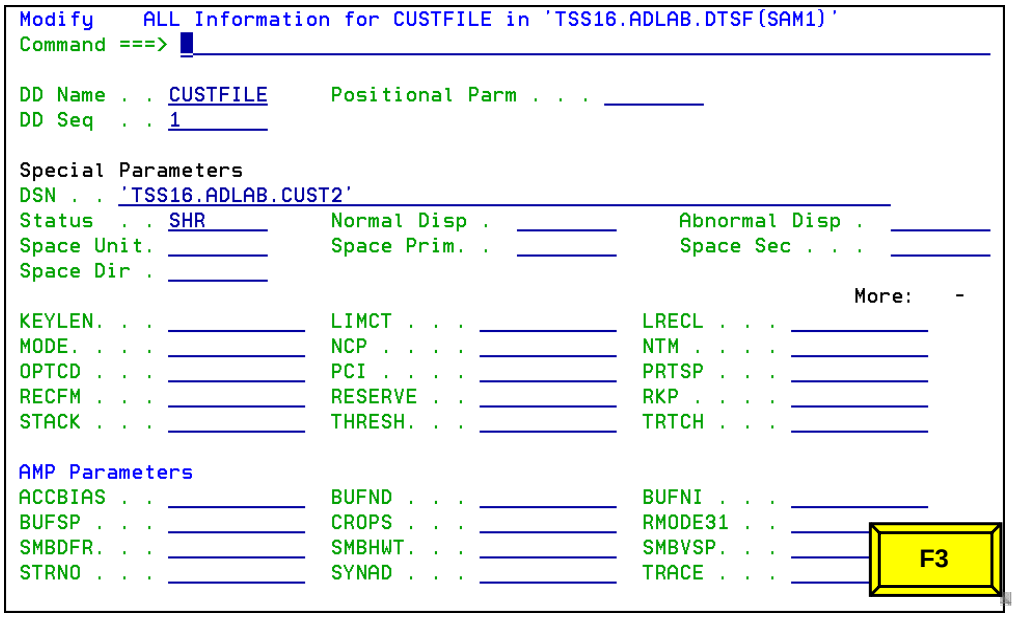
The second 'All information' panel is displayed. Notice that DCB information can be specified here. F8 is pressed.

Additional options available from the SX (select all DD information) line command



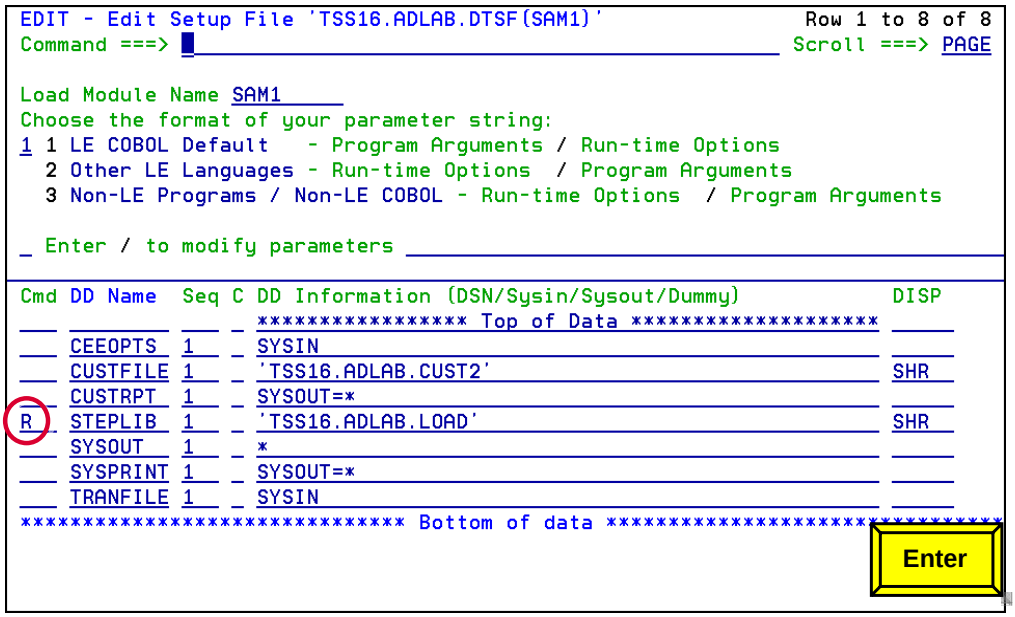
The third 'All information' panel is displayed showing more options. F8 is pressed.

Additional options available from the SX (select all DD information) line command



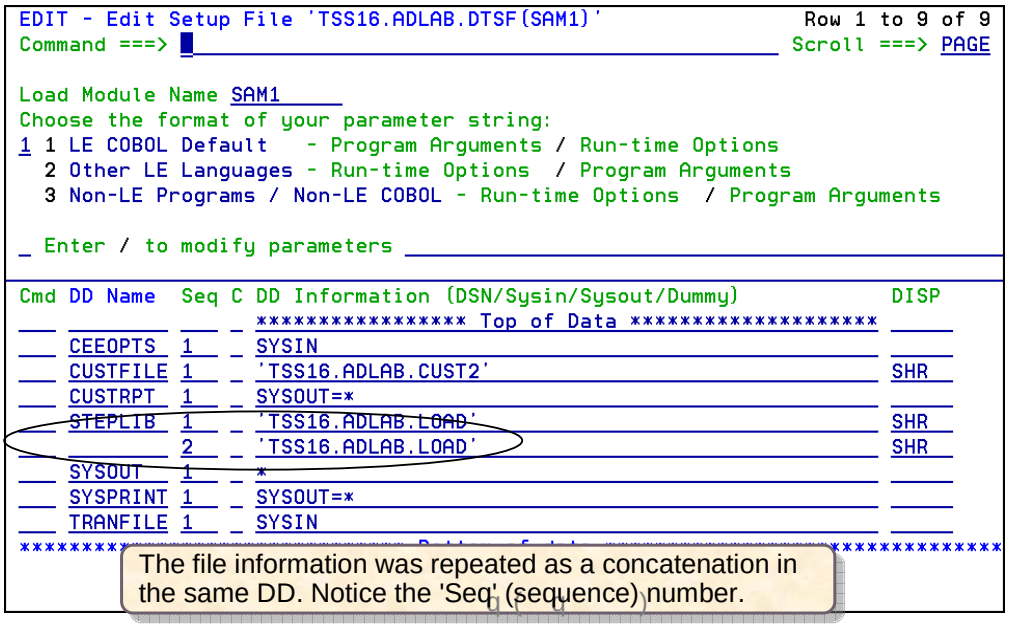
And the fourth and final 'All information' panel is displayed. Specify options as needed for your files. Press F3 to return.

Use an R (repeat) line command to concatenate a file in a DD

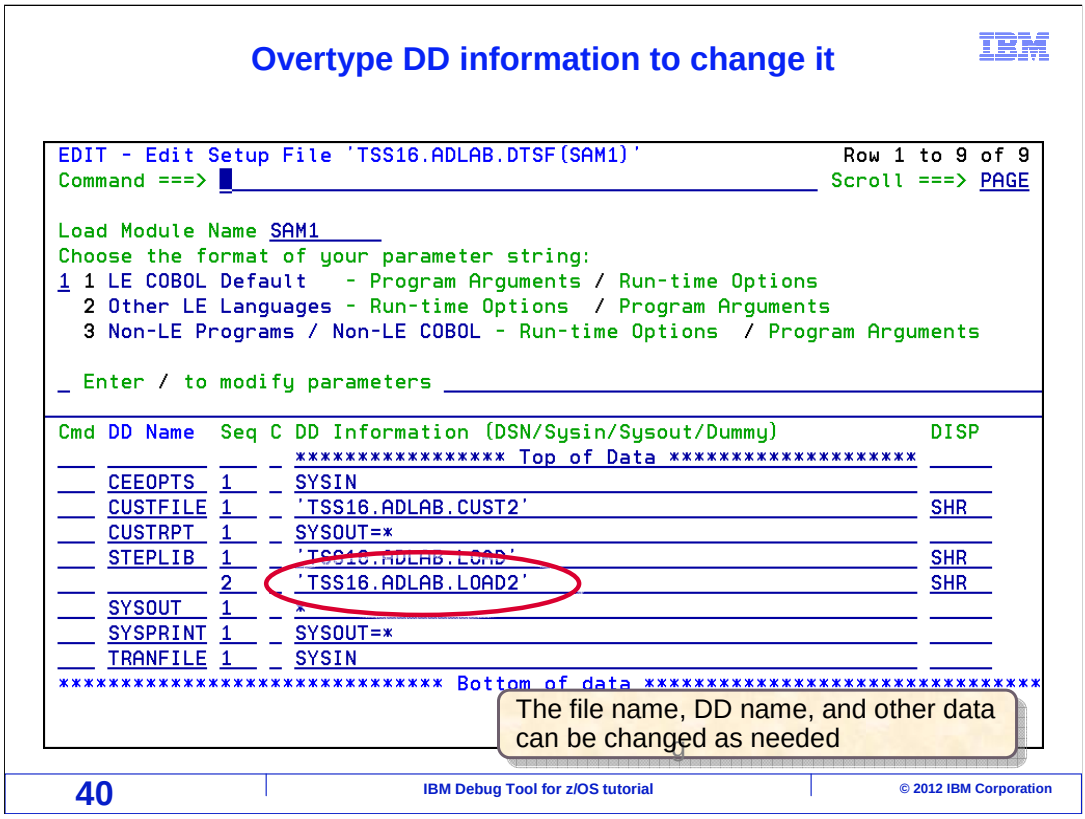


Next, you will see how to repeat a DD statement, and how to code a concatenation. To repeat a DD, type an R line command next to it, and press enter.

Result of the R (repeat) line command



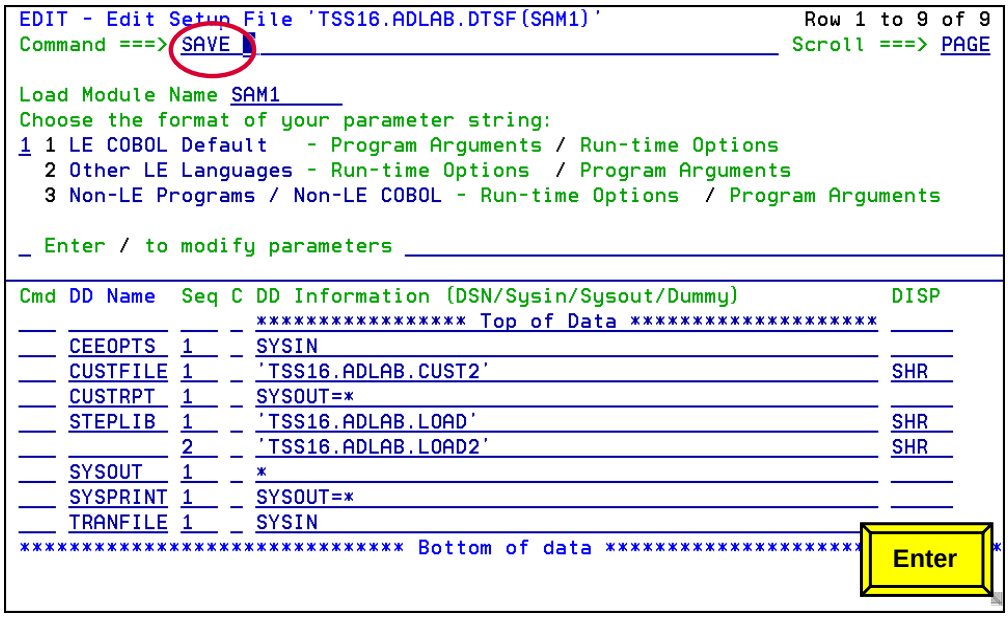
The DD was repeated. Notice that an R line command adds the repeated line as a concatenation of the DD. In this case, the R line command was specified on the STEPLIB DD. Now, STEPLIB is a concatenation of two files.



Now the name of the second file is changed. Notice the that there is a '2' in the sequence field of the second file in the concatenation. You can create concatenations of multiple files in DDs, and you can change the number in the sequence field to change the order of the files in the concatenation.

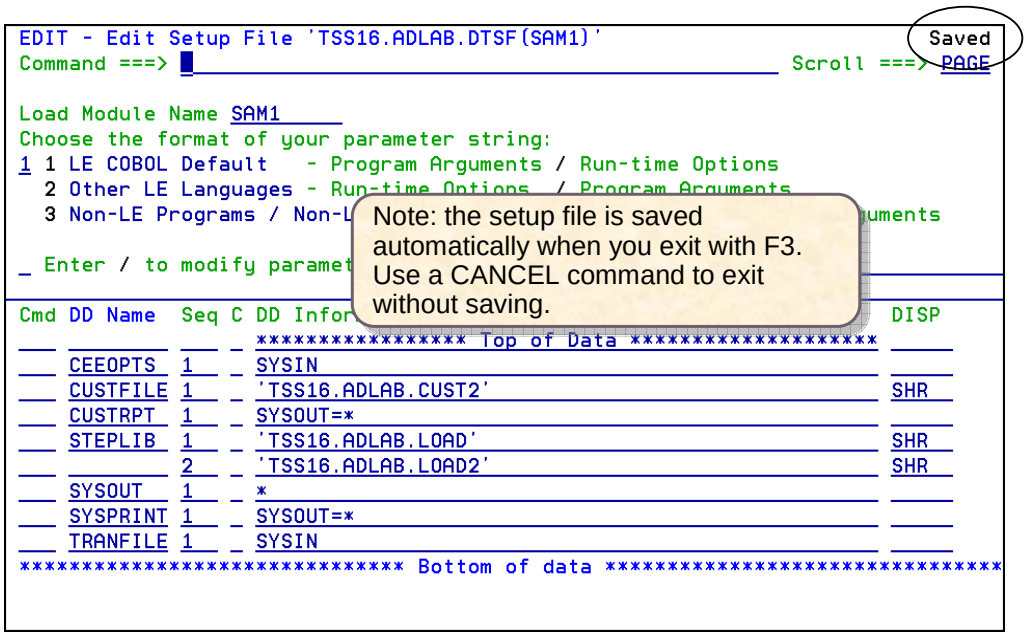
If you do not want the second file to be a concatenation, that is, you want it to be it's own DD, then type a DD name next to the file in the 'DD name field'.

Use a SAVE command to save the setup file



A setup is saved automatically when you press F3 to exit. But you can save a setup file without exiting. Type SAVE on the command line, and press Enter.

The setup file was saved



And the setup file is saved. Once you have created a setup file, you can use it again later when you need it.

That is the end of this section, using the 'Debug Tool setup file' utility to run and debug a batch application in your TSO session.

Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send email feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_DTv12s10DebuggingBatchInTSO.ppt

This module is also available in PDF format at: ../DTv12s10DebuggingBatchInTSO.pdf

You can help improve the quality of IBM Education Assistant content by providing feedback.

Trademarks, copyrights, and disclaimers

IBM, the IBM logo, ibm.com, z/OS, and zSeries are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of other IBM trademarks is available on the web at "Copyright and trademark information" at http://www.ibm.com/legal/copytrade.shtml

Other company, product, or service names may be trademarks or service marks of others.

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS OR SOFTWARE.