**Xpeditor IMP Points**

* Press **F9** for line by line execution
* Press **F12** for break point execution.

For placing a break point press **A** in the line to create a breakpoint.

* Press **D** to delete a break point in the break point line
* Press **QUIT** in the command line to come out of debugging
* **K** WS-TO-DATE to find the variable value
* **RETEST** restart the test from the beginning
* **For Backtracing** use **Monitor** in command line, from the break point where you want to activate backtracing say first breakpoint then type **REVERSE** in command line and press F12.
* **DEL ALL** to delete all breakpoints
* **T**o give breakpoint when variable changes

EG: **WHEN WS-TO-DATE**

* **Enter S to skip statement**
* **WS** we can see entire working storage vars, command is like display (**D**) in Interest.

**Q. What is DDIO file in Xpeditor?**

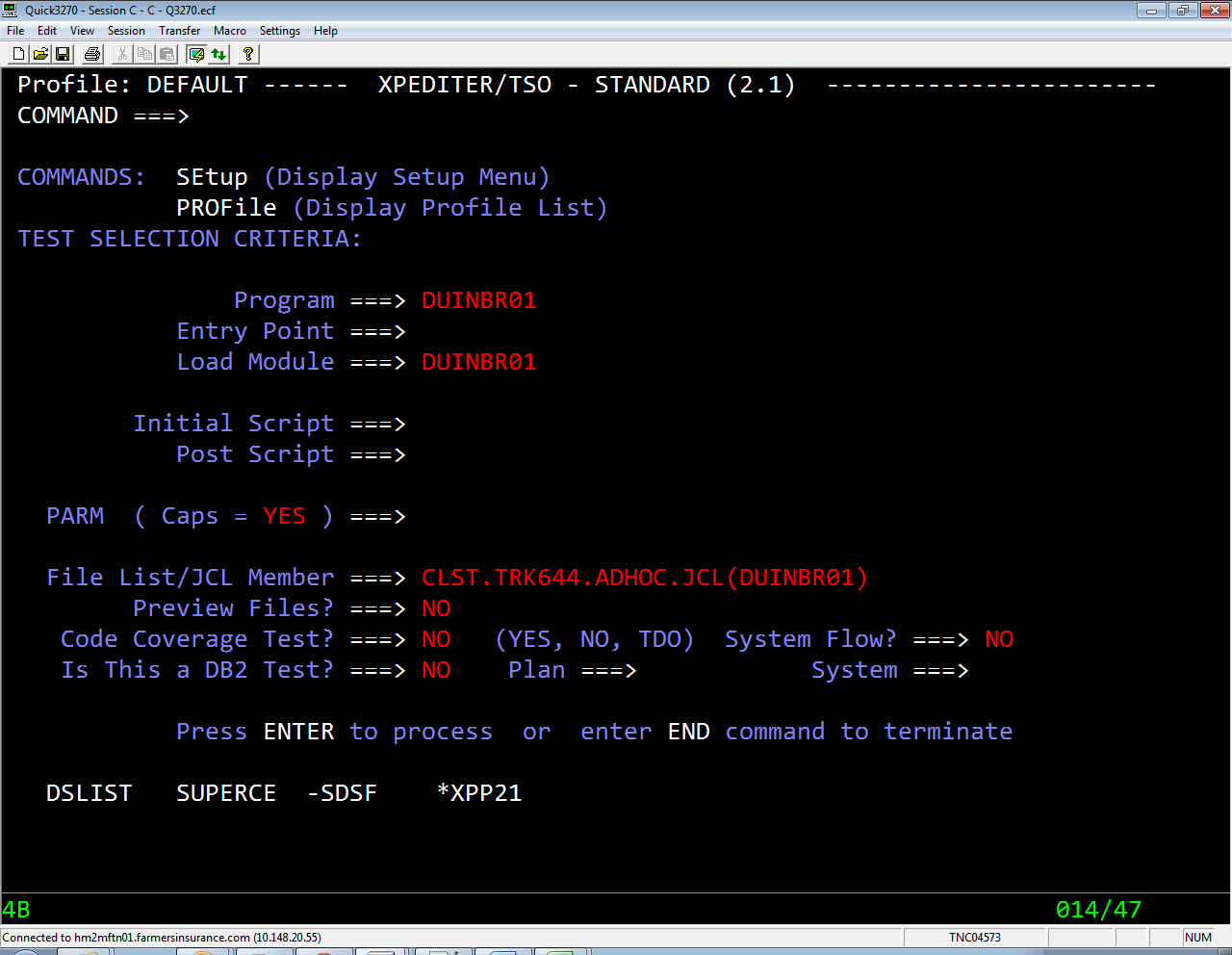
**ANS.**

**DDIO file** is special kind of VSAM **file** which gets creating while compiling your program with **Xpeditor** option. This **file** store the source listing information into it. This **file** can be then used by **Xpeditor** during compile time to use the source listing information stored in it.

**Debugging option in Xpeditor**

START BB;;2(TSO)

Enter the details of the pgm, load and jcl as below.



**Note:** compile the cobol code using the following jcl

**JCL**

//TRK644AN JOB TRUCK,'ENDVRADD TRK644 ',SCHENV=PRC1,

// MSGCLASS=Y,CLASS=X,REGION=0M,

// NOTIFY=&SYSUID,TIME=1440 TYPRUN=HOLD

//\*------------------------ C1SB3000 V15 -----------------------------\*

//\* ACTION SCL GENERATED BY TRK644 ON PRJ1 2015/04/22 02:11

//\*-------------------------C1SB3000----------------------------------\*

//\*

//NDVRSCL EXEC **NDVRSCL**

//BSTIPT01 DD DSN=CLST.TRK644.SCL(ADDB),DISP=SHR

**SCL-** CLST.TRK644.SCL(ADDB)

ADD ELEMENT **'DUINBR01'**

FROM DSNAME 'CLST.TRK644.ADHOC.PGMS'

TO ENVIRONMENT 'CPRTEST' SYSTEM 'TRUCKA' SUBSYSTEM 'TRUCKA'

TYPE 'COBB'

OPTIONS CCID 'TE' COMMENTS "test" OVERRIDE SIGNOUT UPDATE

PROC GROUP **'GCOBB02'**

.

**COMMAND** & **ACTION**   
  
**AFTER** Breakpoint after execution of line   
**BEFORE** Breakpoint before execution of line   
**BOTTOM** Scrolls to bottom of currently displayed data   
**COUNT** Sets execution counters to gather test coverage statistics   
**DELETE** Removes all XPEDITOR commands (e.g. breakpoints)   
**DLEFT** Scroll data in Keep/Peek window to left- can specify amount   
**DRIGHT** As above to the right   
**END** Terminates current function and returns to previous screen   
**EXCLUDE** Excludes data lines from displaying in the source   
**EXIT** Terminates the current test session   
**FIND** Searches for character strings, data names and COBOL structures.   
**GO 1** Walks through code (equivalent PF9)   
**GO** Goes to next breakpoint (equivalent to PF12)   
**GOBACK** Changes the program logic and returns higher level module   
**GOTO** Repositions the current execution pointer   
**HELP** Displays info about error message or gives tutorial info.   
**IF** Establish a conditional expression in a block of inserted lines   
**INSERT** Temporarily insert XPEDITOR/TSO commands in the program   
**INCLUDE** Include command executes a predefined test script member   
**KEEP** Displays the values in a chosen field in Keep window   
**LEFT** Scrolls source listing to left by specified amount   
**LOCATE** Scrolls to particular line number.   
**MEMORY** Displays memory for a specified location   
**MONITOR** Records program execution in a buffer.   
**MOVE** Changes the contents of program variables   
**PAUSE** Sets a pause breakpoint within inserted lines or commands   
**PEEK** Displays values of program variables.   
**RESET** Restores excluded lines in source screen   
**RETEST** Begins a new test of the same program   
**REVERSE** Reviews the execution path that led to the current breakpoint.   
**RIGHT** Scrolls the source to the right by a specified amount   
**SET** Overrides XPEDITOR/TSO defaults.   
**SHOW** Displays breakpoints, diagnostic info or SKIP Temporarily bypasses the execution of a statement   
**SOURCE** Changes the module shown on the source display during Interactive debugging   
**TOP** Goes to the top of the data   
**UP** Scrolls to the top of data   
**WHEN** Indicates when a specified condition is true or when program variable changes value.   
**WS** Displays Working storage   
**ACCEPT** Accepts the data from I/P file or from Instream data

Read more: <http://ibmmainframes.com/about5140.html#ixzz4bZYx9FRV>

Give unconditional break point ie even if the transaction is running in passive mode the breakpoint is called.

