BCI433 Lab 3B (updated Winter 2021)

**Writing an interactive RPGLE screen program**

**Lab objectives:**

* **Use Case structure to solve an application problem**
* **Use the Debug option to check a program’s logic**

**Lab Requirements:**

**Show compiler listing for COVIDRPG (with Lab3B logic including Case structure)**

**Successfully run COVIDRPG (with Lab3B logic including Case structure)**

Start an RDi session

Start a ‘Green Screen’ (emulator) Session.

**Using** **Rational Developer for i (RDi):**

**Part A**

**Objectives:**

Case structure is a good choice to solve application logic problems. Here is an example of that:

SELECT;

When Day = 1;

DayName = ‘Monday’;

When Day = 2;

DayName = ‘Tuesday’;

When Day = 3;

DayName = ‘Wednesday’;

When Day = 4;

DayName = ‘Thursday’;

……

ENDSL;

If Day is a 1, then the first test would be done, the DayName would be set to ‘Monday’ and control would go to ENDSL. The test for Day = 2 would not be performed. If Day is a 7 then seven tests would be done and a DayName = ‘Sunday’ statement would be executed.

If an 8 or 9 was entered in Day you could include an Other clause in place of a When clause at the bottom of this structure before the ENDSL. **Other** is a catch all.

Other

DayName = ‘Invalid Day Number’

ENDSL

Tests done with the select When statement can include <=, >=, <> (not equal to), AND, OR

In class exercise:

You need to figure out how to convert a number to a grade.

The mid term test is a mark out of 100 and is worth 35% of the final grade. The labs are entered as a mark out of 30 and are 30% of the final mark. The exam is a mark out of 100 and is worth 35%. If a student fails either the final exam or the test, they will fail the course. The final grade is stored in a field called NUMGRADE. After this grade has been determined, it needs to be converted to a letter grade LETGRADE.

The following information is used to do this.

A+ = 90 - 100 A = 80 – 89 B+ = 75 - 79 B = 70 – 74

C+ = 65 - 69 C = 60 – 64 D+ = 55 - 59 D = 50 - 54

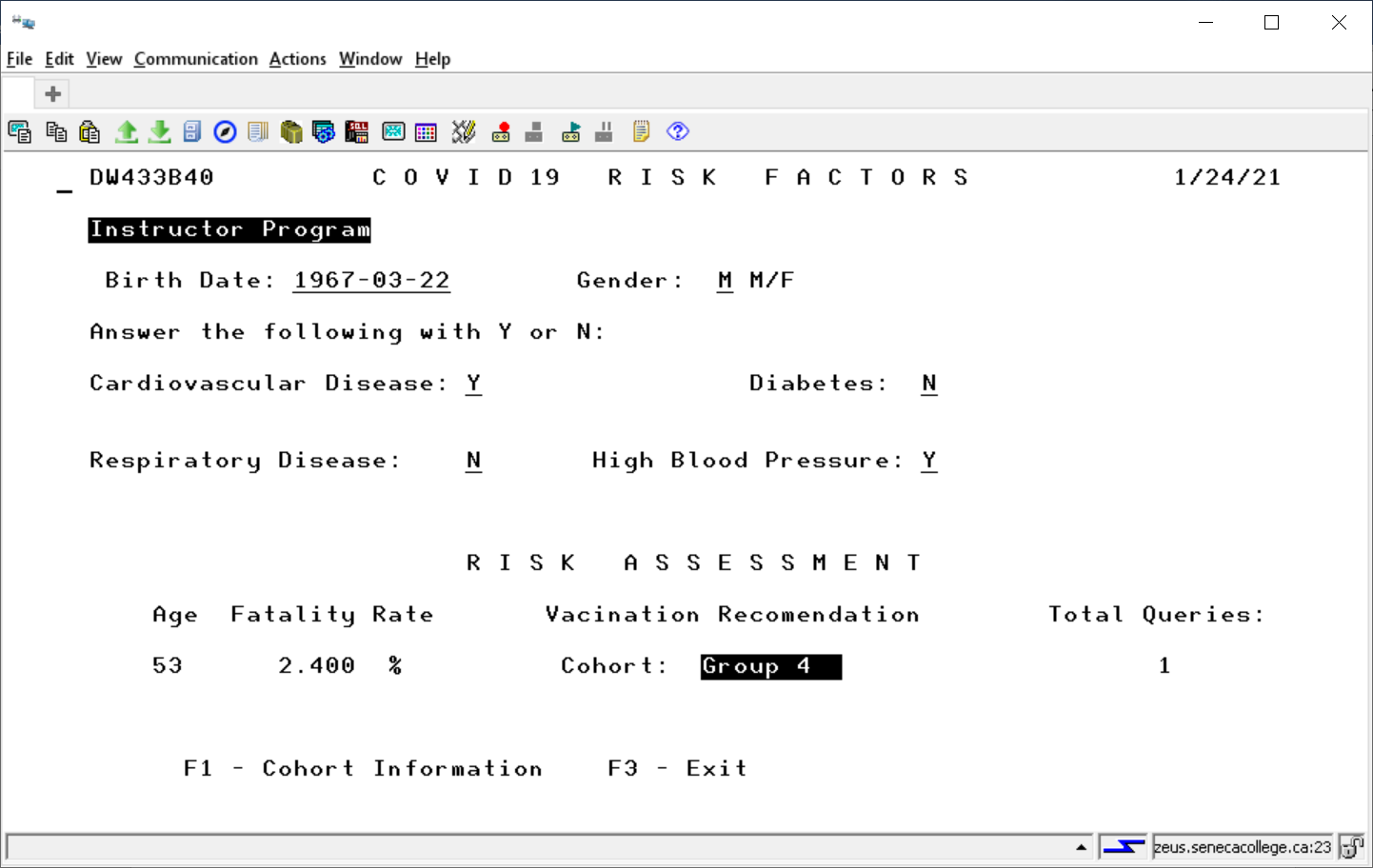
F = 0 – 49

Solution:

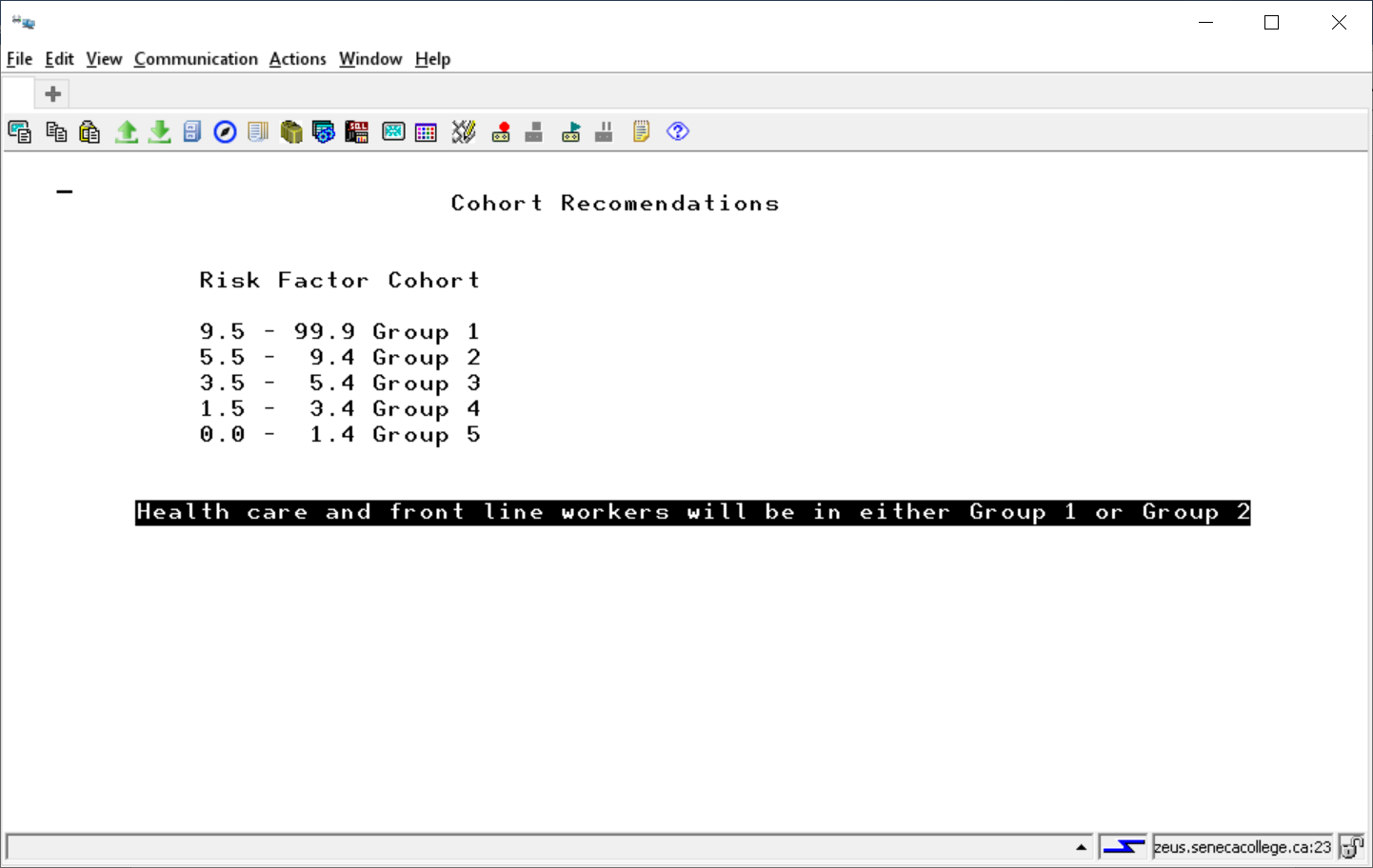
Processing

You have an interactive RPGLE program running from lab 3a with no logic to determine the risk factor, age and vaccination cohort. Apply the information and features below to determine the correct output for completion of lab 3b.

Allow the F1 key to be pressed when at the screen that shows an overlayed record.



Use the information shown on this screen to determine the vaccination group. Have your program show this screen when the F1 key is pressed from the RISKASSMNT screen.



In order to access this screen without coding it, make sure BCI433LIB is always on your library list when compiling or running your program. You can declare a second display file in your program called **COVIDINFO**.

The screen record you want to show is called **CINFO**.

Information not shown above:

Use an AgeFactor subroutine to determine the fatality rates based on age. The fatality rate is stored as 5 digits with 3 decimal positions. So 14.8% is stored as 14.800 - Not 00.148

|  |  |
| --- | --- |
| Age Range | Fatality Rate |
| 80 - 999 | 14.8 % |
| 70 - 79 | 8% |
| 60 - 69 | 3.6% |
| 50 - 59 | 1.3% |
| 40 - 49 | .4% |
| 30 - 39 | .25% |
| 20 - 29 | .23% |
| 10 - 19 | .2% |
| 0 - 9 | .1 |

Use an OtherFactors subroutine to determine modifiers that increase the Fatality Rate. These can be added directly to the fatality rate.

|  |  |
| --- | --- |
| Additional Risk Factor | Increase by or decrease by |
| Cardiovascular Disease | + .8 |
| Diabetes | + .6 |
| High Blood Pressure | + .3 |
| Respiratory Disease | + .4 |
| Female | -.05 |

Do not test both the upper and lower limits of a range – you only need to test one or the other depending on your approach. Testing both limits is wasting CPU time. Not a big problem with this application, but may be a factor when processing millions of records. Make sure your solution is efficient.

**Disclaimer – Some research went into the numbers used, but note these numbers may be way off the real numbers, so we will view them as unreliable.**

The best way to check the behaviour of the program is to run the instructor version.

ChgCurLib BCI433LIB

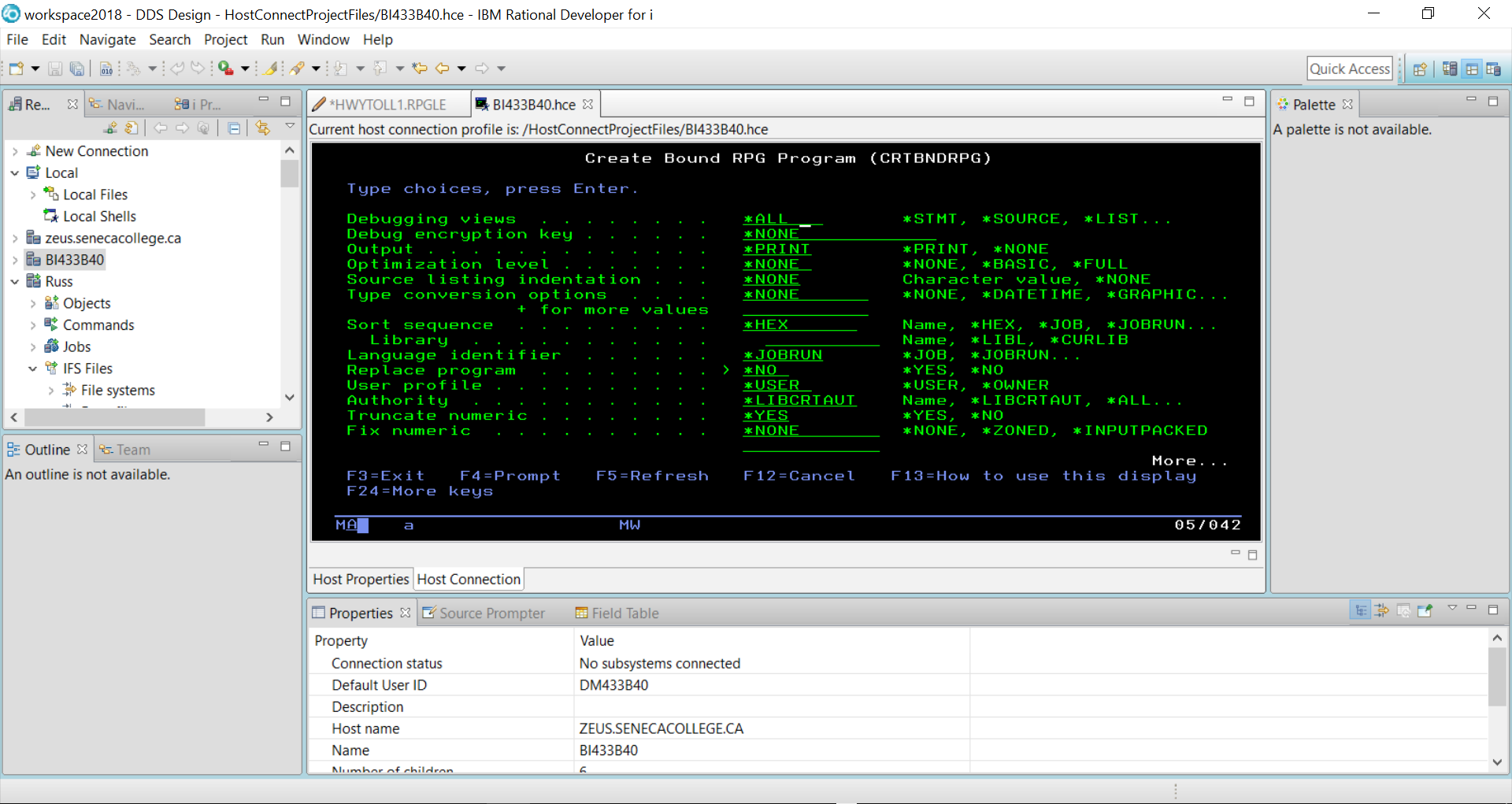
Call COVIDRPGB

ChgCurLib (Back to your library)

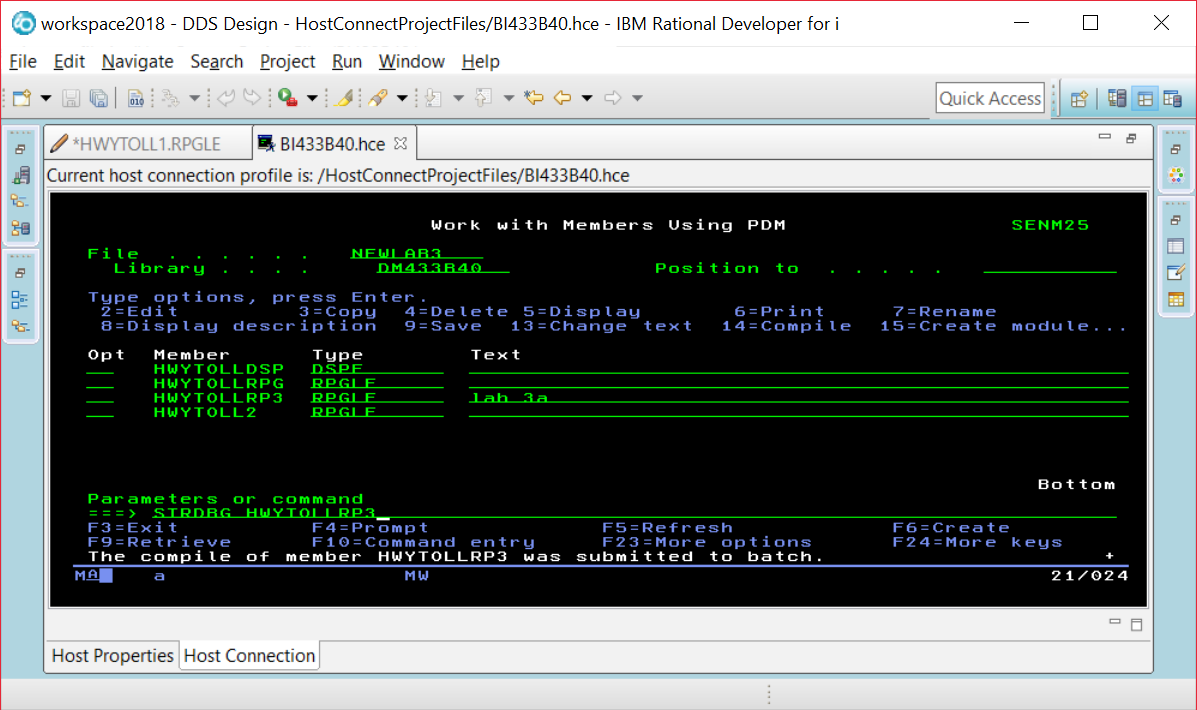
Your program should get similar results. In order to test your program, your professor will try a few scenarios, scan your PDF listing to see if there are any obvious errors. Your listing can be printed by a PC printer and should have utilized the CALL STRJOB command before compilation so your id and name appear at the bottom of each page.

Using Debug With an Interactive RPGLE Program

Green screen compile requires an option to be changed. Debugging views should be set to all. RDi compile does this by default.

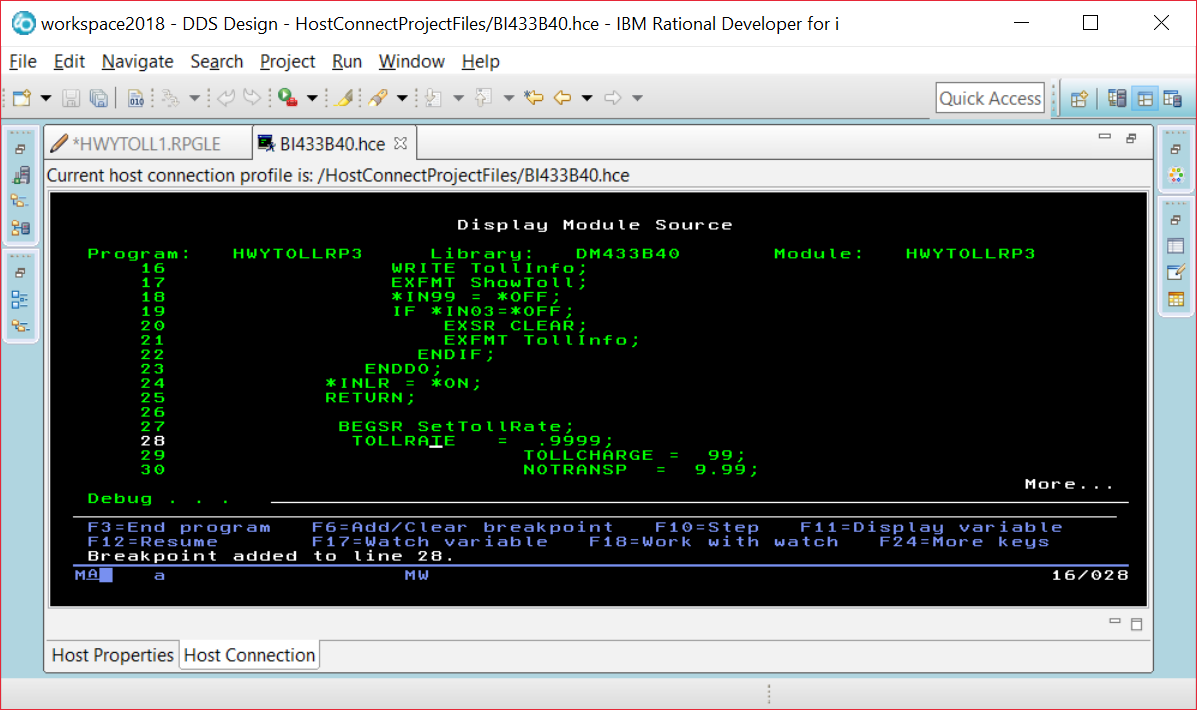


The STRDBG command allows you to set a breakpoint in the program when it is running.

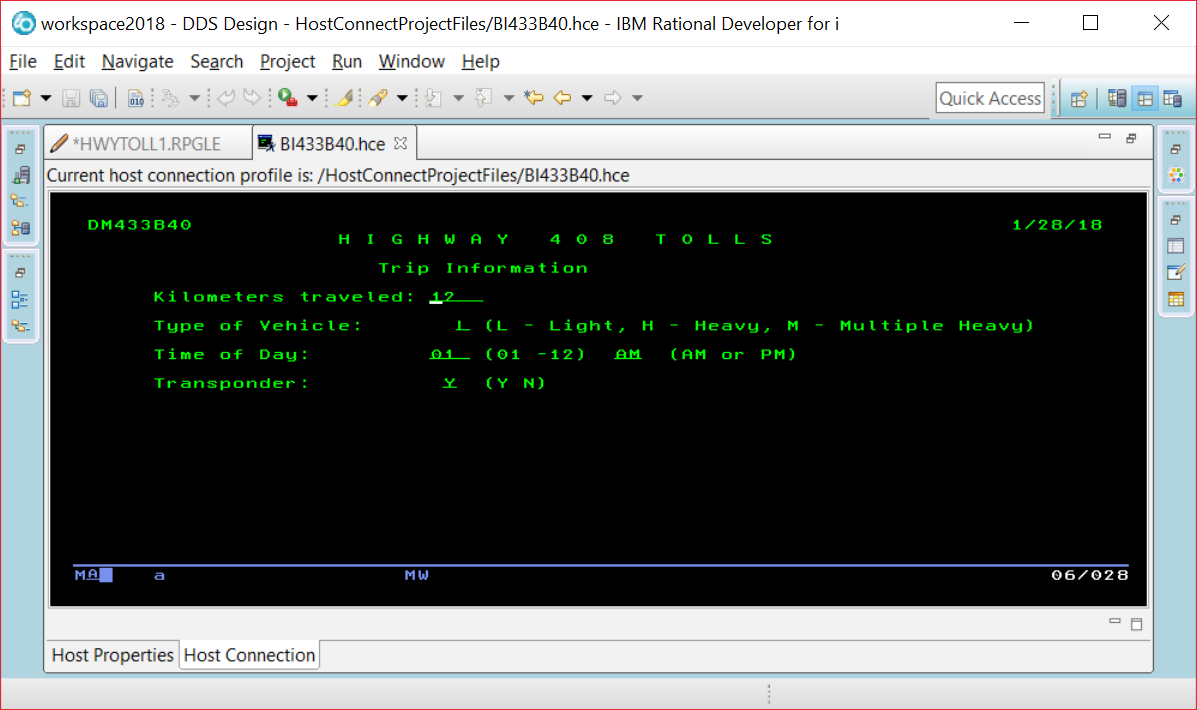


If you are already in DEBUG mode, this command will not work. Just enter the ENDDBG command to exit from DEBUG mode and then enter the STRDBG command with your program name.

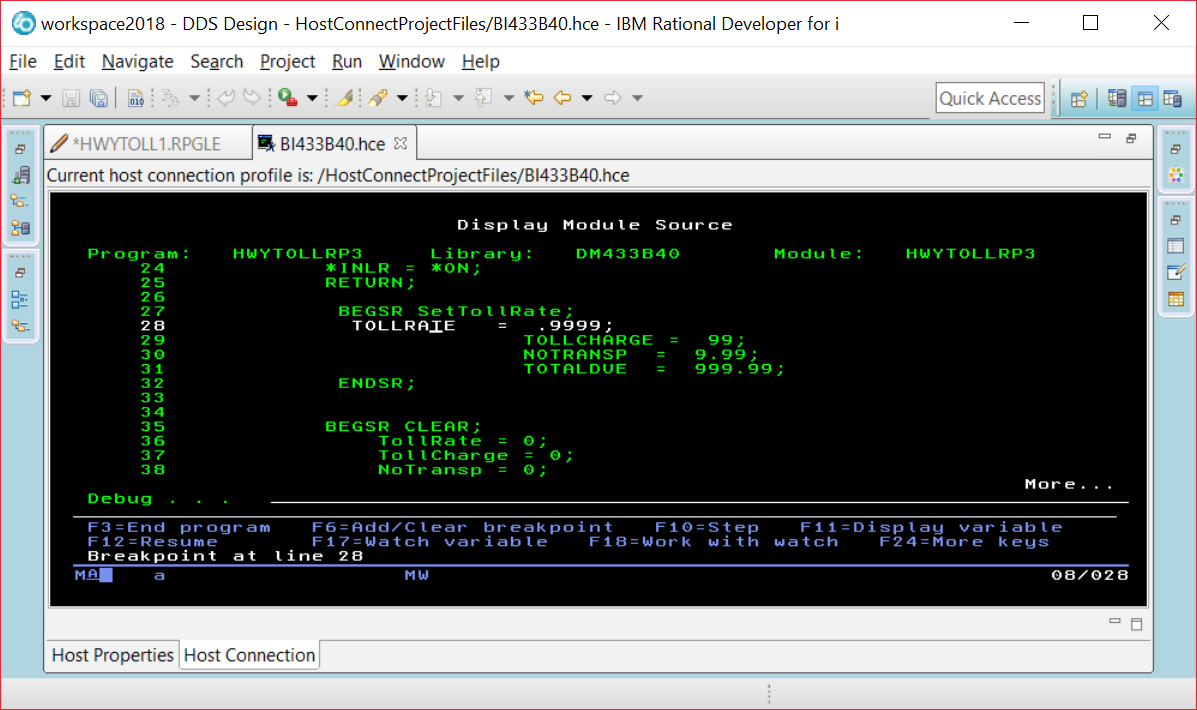
Page down and put your cursor on the line where you want a breakpoint, and press F6.



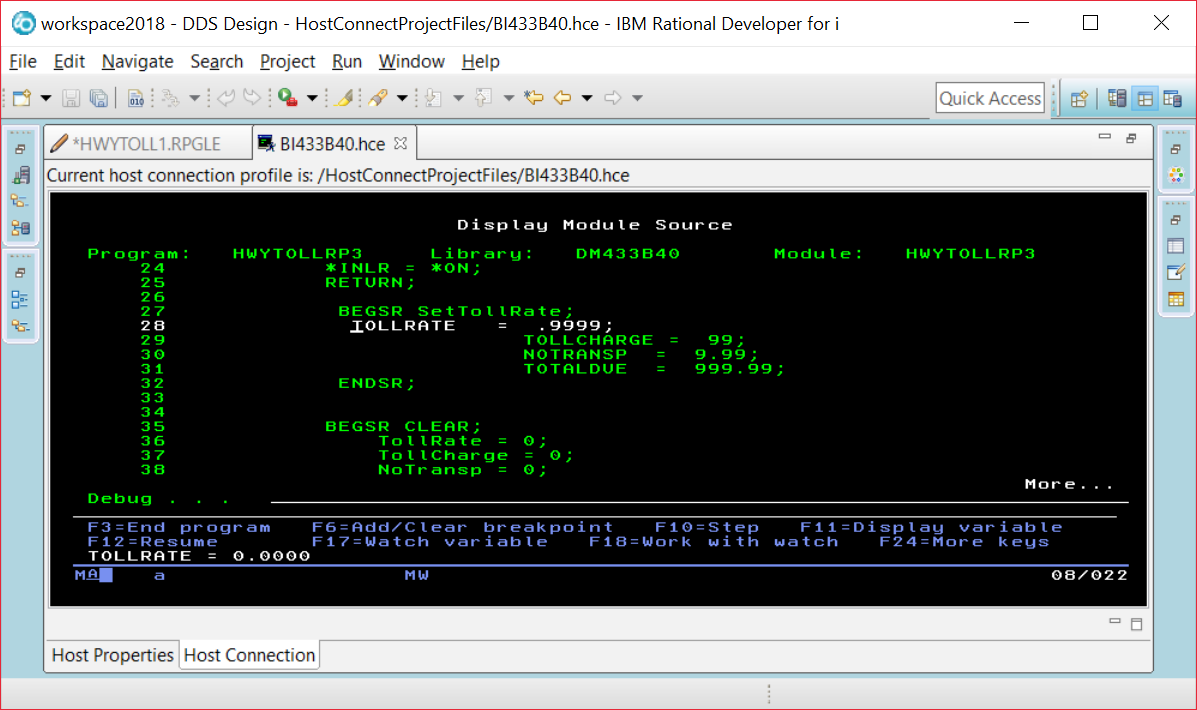
When you run the program you will see screens the program displays before the breakpoint. Our breakpoint was after the EXFMT line, so we see this screen and can enter data into the fields.



The breakpoint is reached and this line has not been executed yet. So TOLLRATE would still be set at 0.

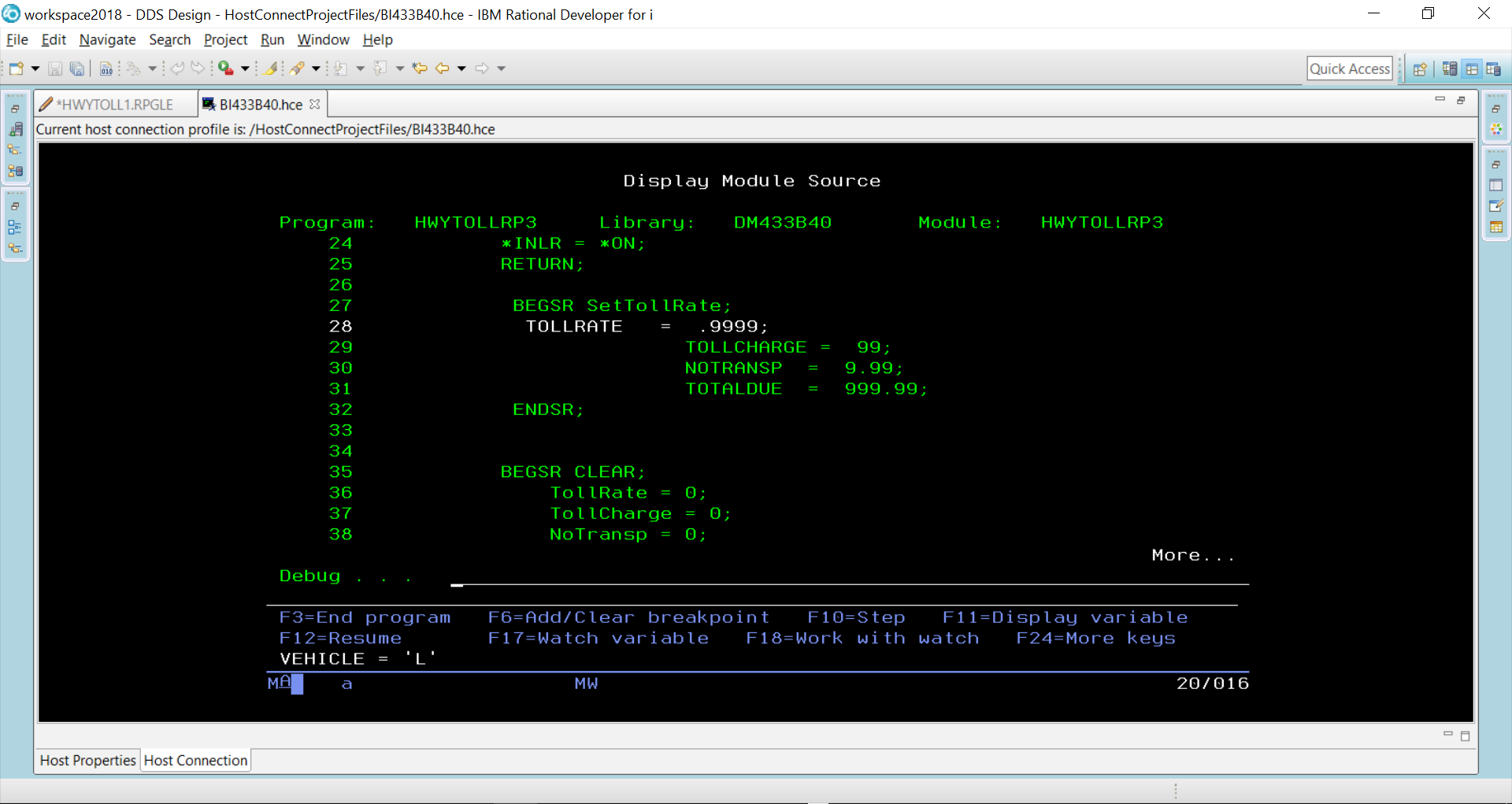


If you put your cursor on the TOLLRATE field, you will see what is currently stored in the field.



Instead of using F11, you run the command EVAL VEHICLE command at the prompt and be able to find out what is in a field that is not currently being displayed in a line of code on your screen.

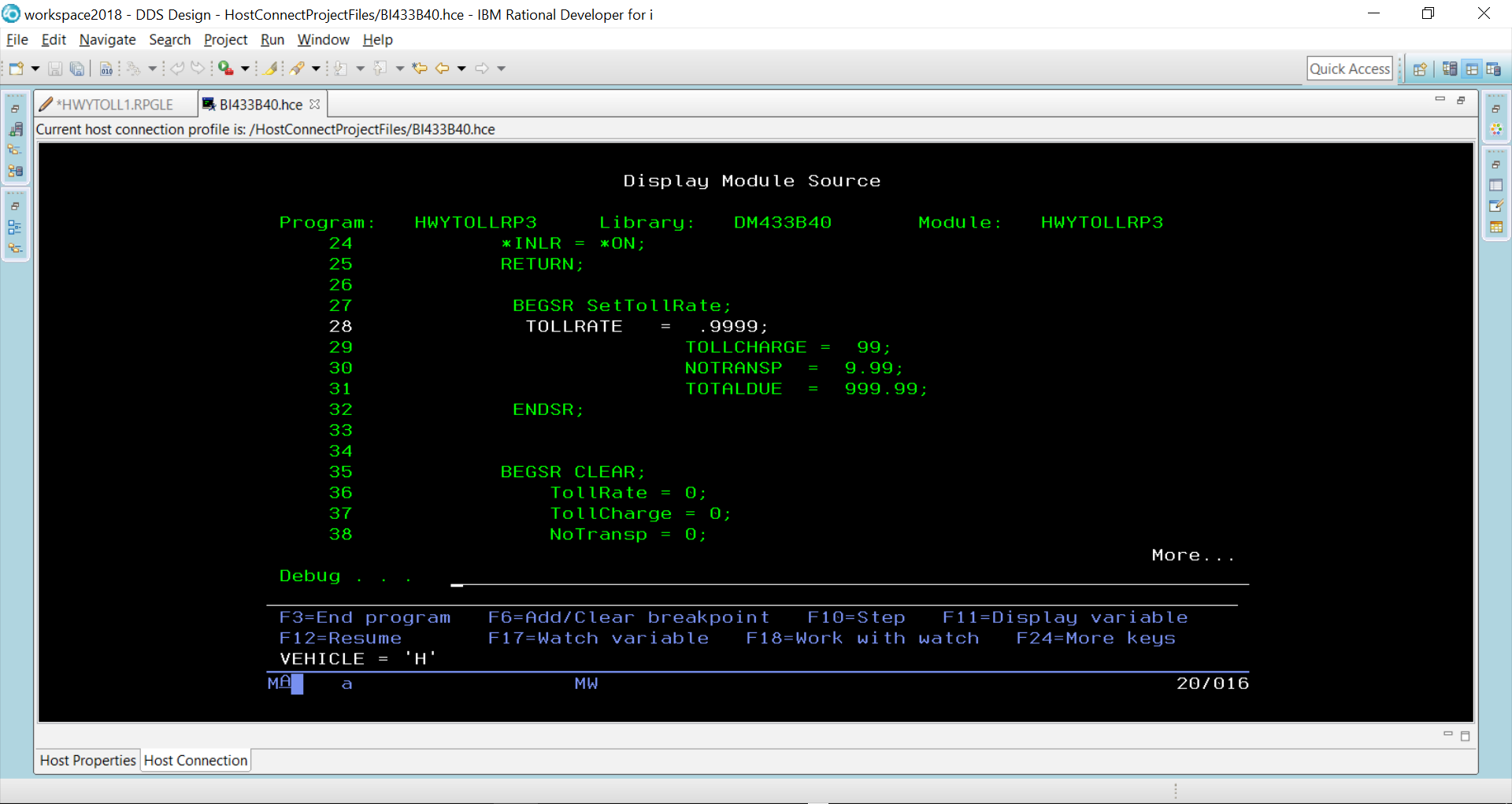
EVAL VEHICLE



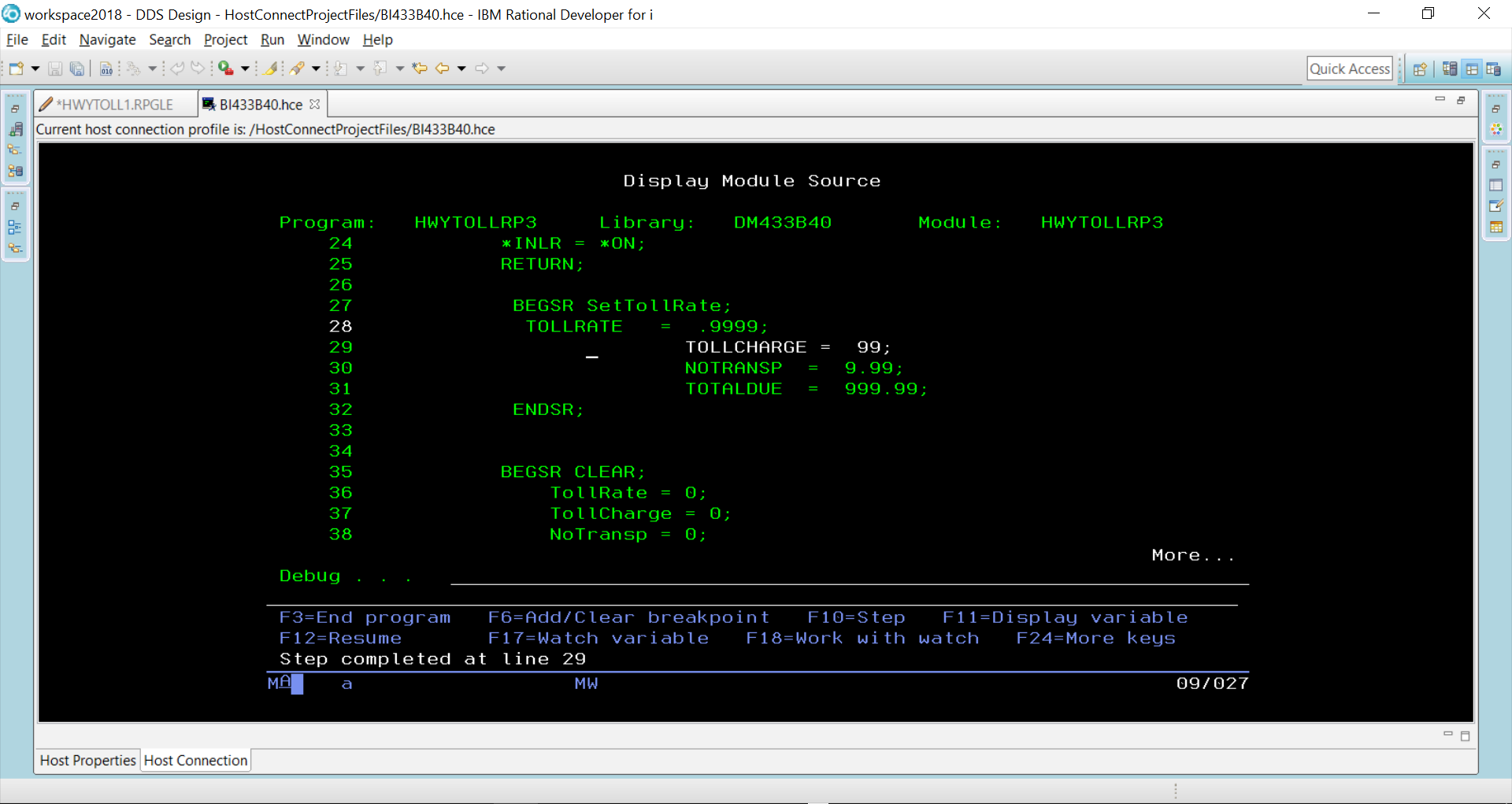
You can alter the contents of a field while your program is running.

EVAL VEHICLE = ‘H’

EVAL VEHICLE



F10 is pressed to step through each line of code



TollRate would now be set at 9999 because line 28 has executed.

When F12 pressed – the rest of the code executes until we get to where the two screens are made available and the program is paused.

