BCI433 Lab 3B (Summer 2022)

**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 WATERBILL (with Lab3B logic including Case structure)**

**Successfully run WATERBILL (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 for Lab3B

You have an interactive RPGLE program running from lab 3a with no logic to determine the charges for water usage. Apply the information and features below to determine the correct output for completion of lab 3b.

Text

Description automatically generated

A rate is determined but is not shown on any screens. Search on how to define a three digit with two decimal places (9.99) free format standalone field. Do a google search for “**rpgle standalone field free format**” Don’t waste a lot of time on how to do this – if you need help, ask your instructor. Refer to this in your program as RATE.

With the first screen example the **rate** was determined to be 2.66. That rate is applied to the cubic metres of water used to determine a water charge of $151.62. The rate is not shown on the screen allowing the field size and type to be brought in at compile time, so you need to use a standalone field to define it.

Calculating Charges:

Water Rate

Customers are usually rewarded with discounts when they buy more of something. Since water is a precious resource, customers are not rewarded for using larger quantities – their rate is increased.

|  |  |
| --- | --- |
| Cubic Metres Usage | Rate |
| 76 - 999 | 2.85 |
| 60 - 75 | 2.77 |
| 30 - 59 | 2.66 |
| 16 - 29 | 2.54 |
| 0 - 15 | 2.3 |

Any customers using over 100 cubic metres of water are notified that they will be visited by an inspector and may be fined for overuse unless they have an exemption.

Sewer Rate

the sewer rate is $2.76 per cubic metre.  No allowance is made for water used outdoors for pools, watering, washing etc.

Capital Cost Recovery Fee is based on zone. This fee is used to support future costs with infrastructure like replacing pipes and extending service. The neighbourhood zone is based on the current infrastructure state of the zones. Some zones pay a higher amount.

Zones A,B and C charges 18.75. Zones E and F charges 8% of the water and sewer total and Zones G and H charge 5.5 % of the water and sewer total.

If you pay the bill early a 4.5% discount is applied to your total owing

One cubic metre is equal to 219.969 gallons (Imperial gallons). Only show the conversion when both screens are being displayed.

This screen shows a usage amount greater than 100 cubic meters and the associated warnning message that appears when that is the case.

Text

Description automatically generated

If the F4 key is pressed, then a screen will appear that provides information on water usage.

Graphical user interface, text, application, email

Description automatically generated

The user has the option to exit the program or enter more water usage scenarios to see the charges.

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

ChgCurLib BCI433LIB

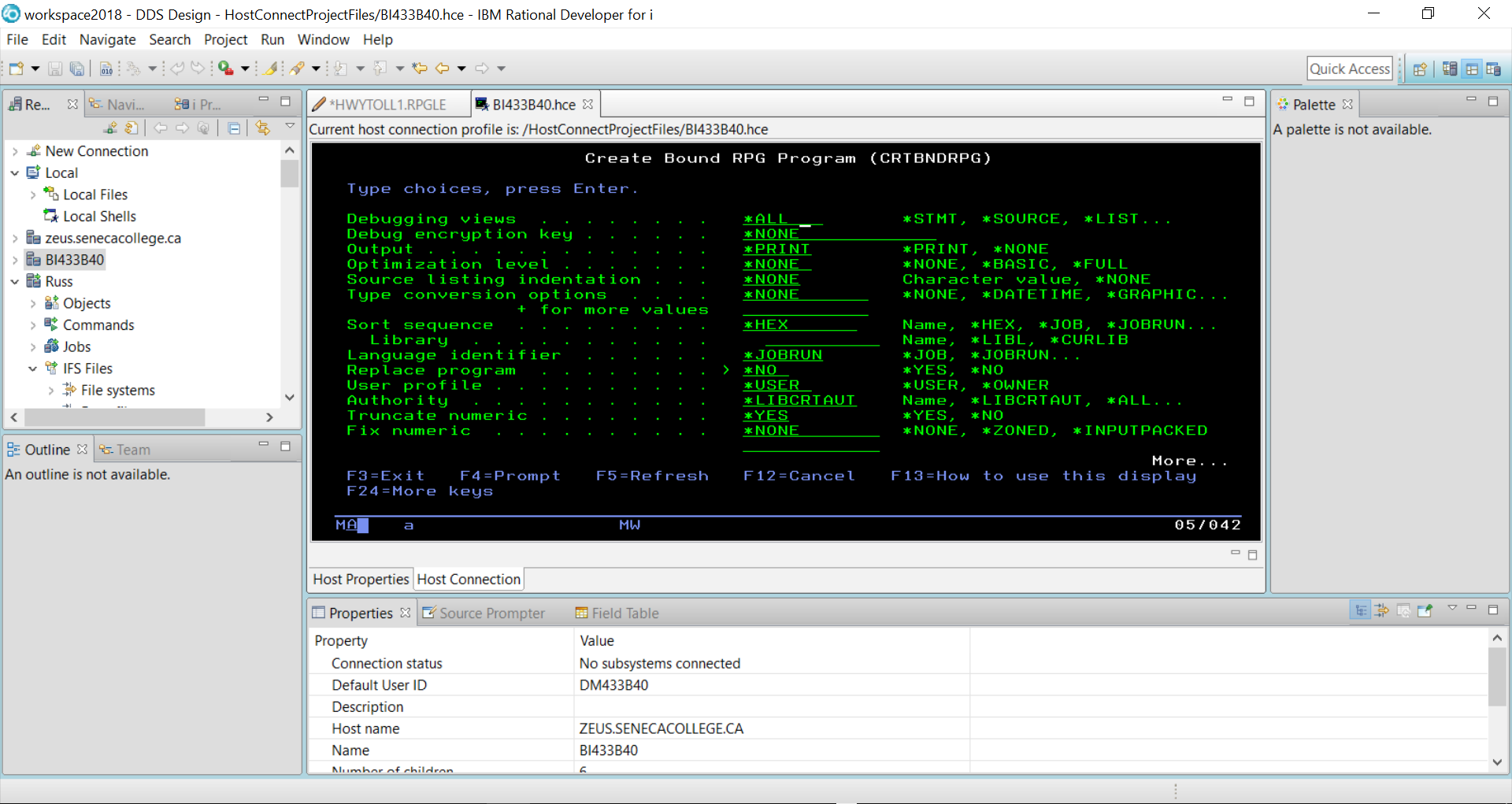
Call WATERBILLB

ChgCurLib (Back to your library)

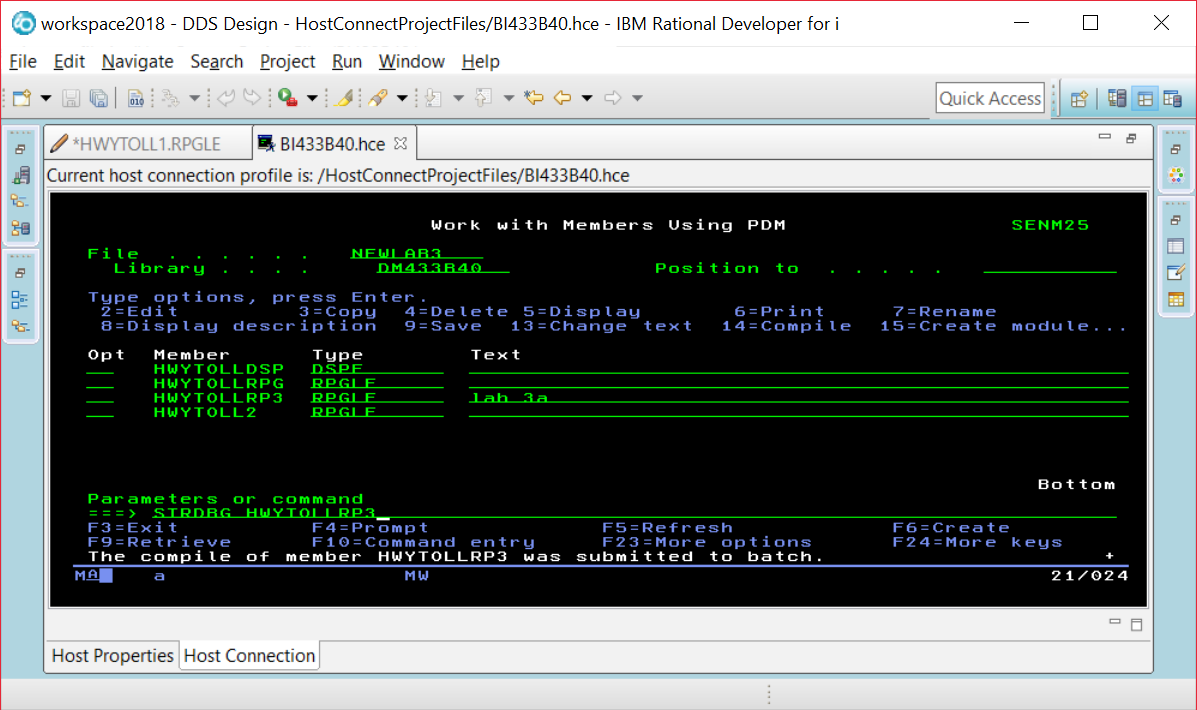
Your program should get similar results. In order to test your program, your professor will try a few scenarios, scan your PDF listing to see if there are any obvious errors. Your listing can be printed by a PC printer and should have utilized the CALL STRJOB command before compilation so your id and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

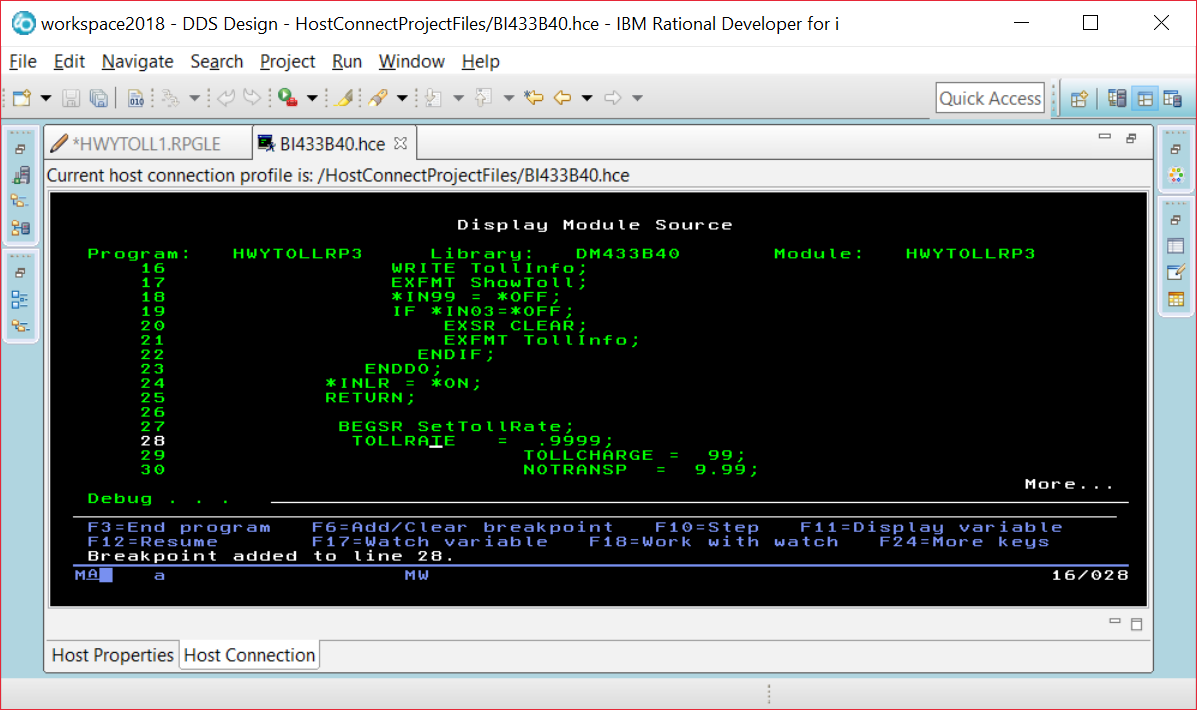


The STRDBG command allows you to set a breakpoint in the program when it is running.

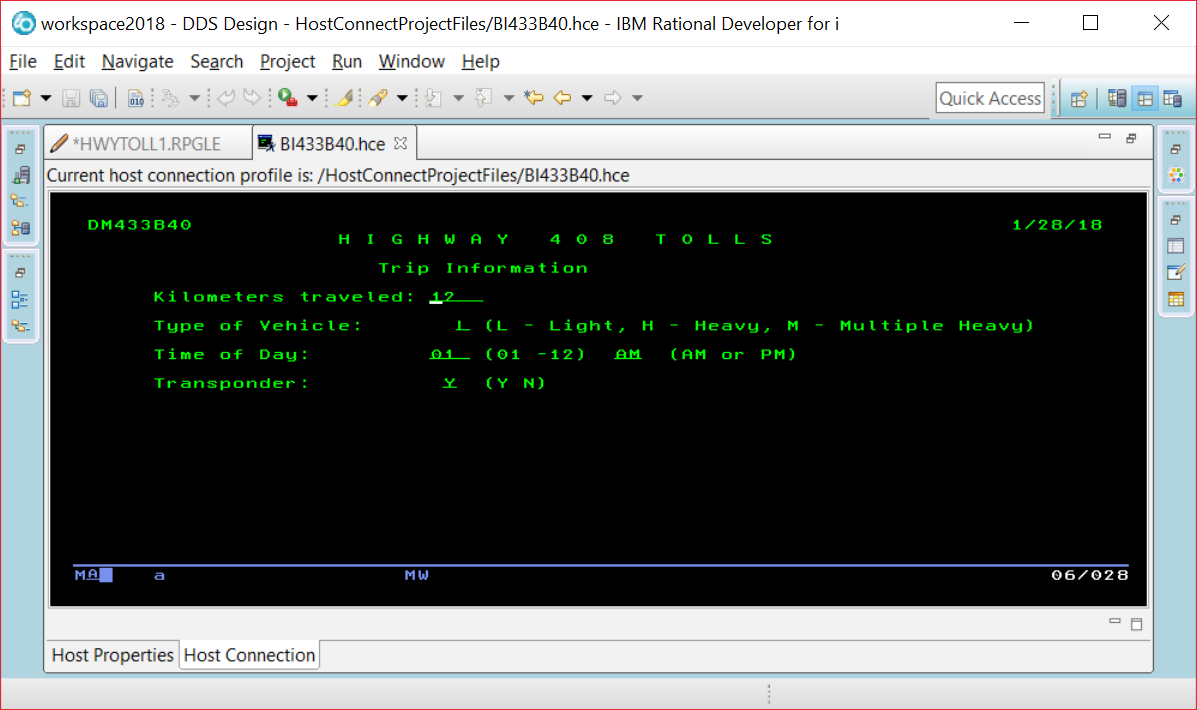


If you are already in DEBUG mode, this command will not work. Just enter the ENDDBG command to exit from DEBUG mode and then enter the STRDBG command with your program name.

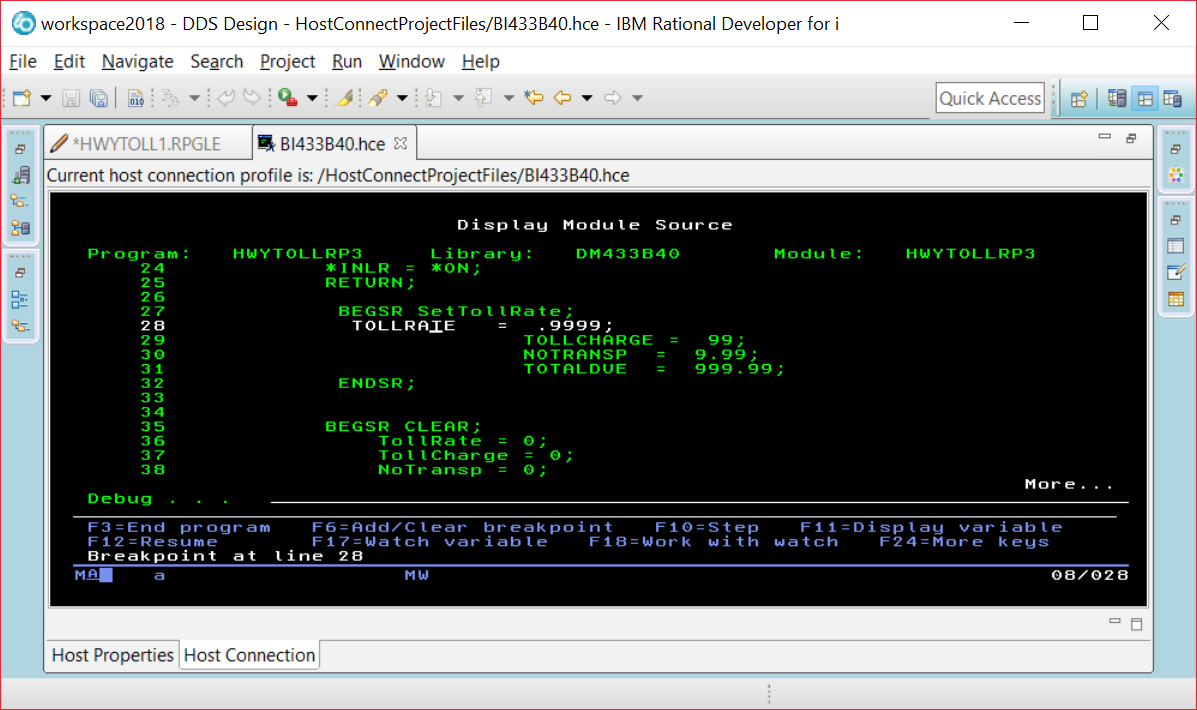
Page down and put your cursor on the line where you want a breakpoint, and press F6.



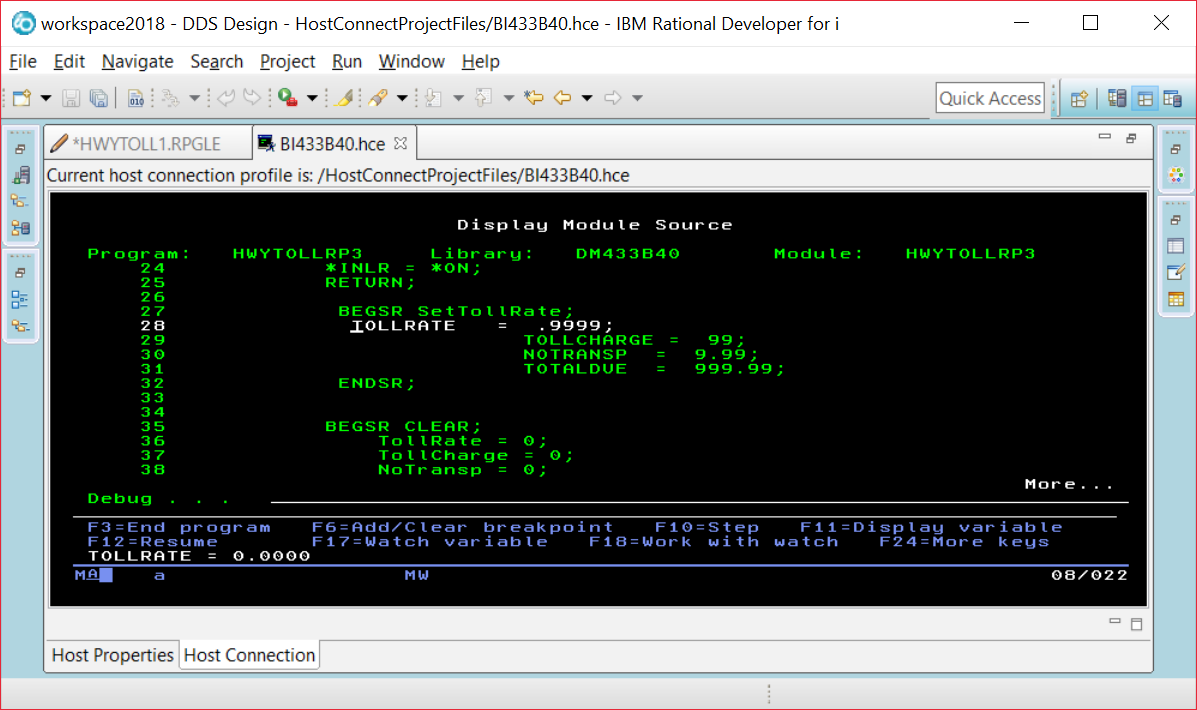
When you run the program you will see screens the program displays before the breakpoint. Our breakpoint was after the EXFMT line, so we see this screen and can enter data into the fields.



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

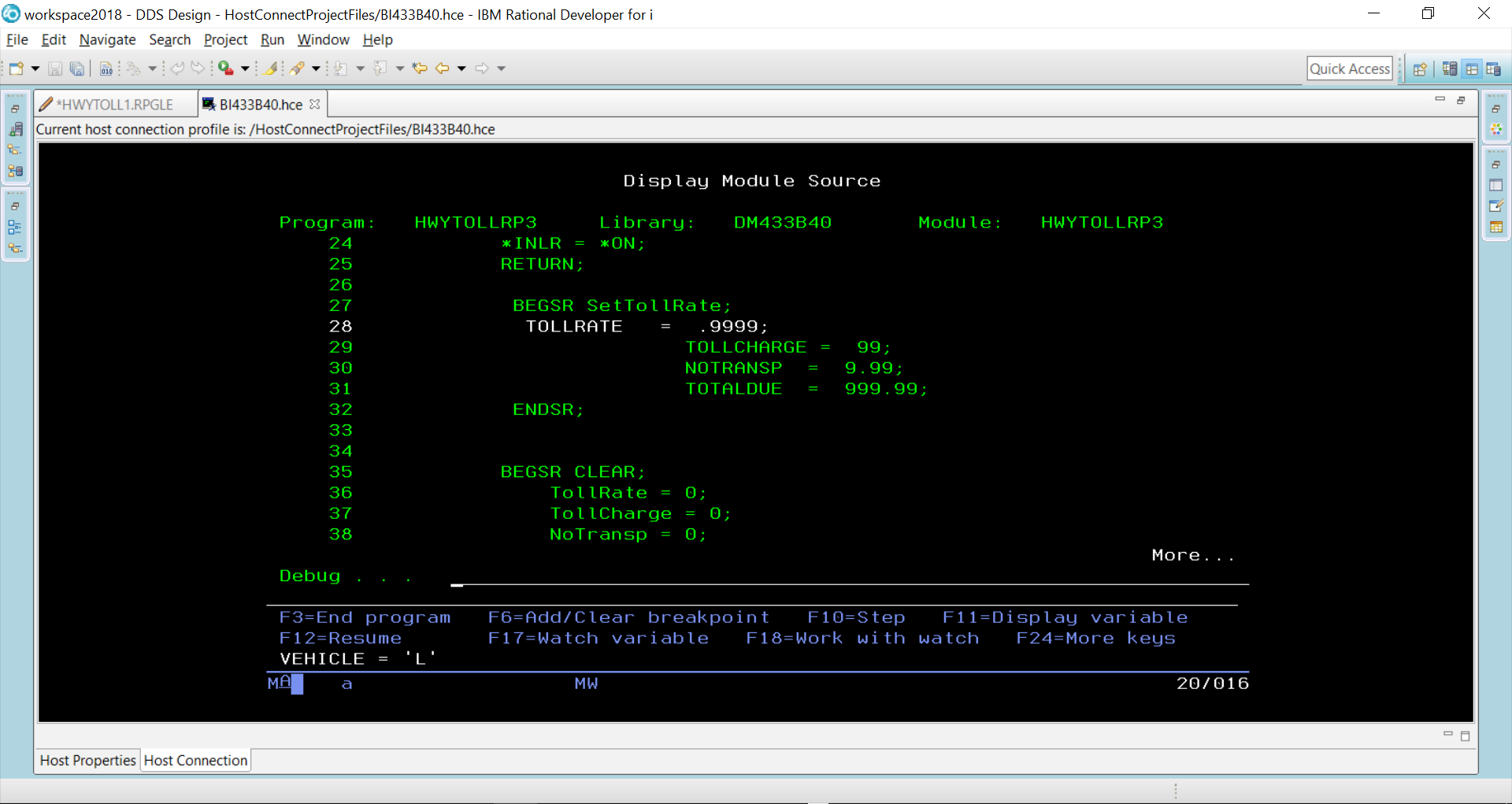


If you put your cursor on the TOLLRATE field, you will see what is currently stored in the field.



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.

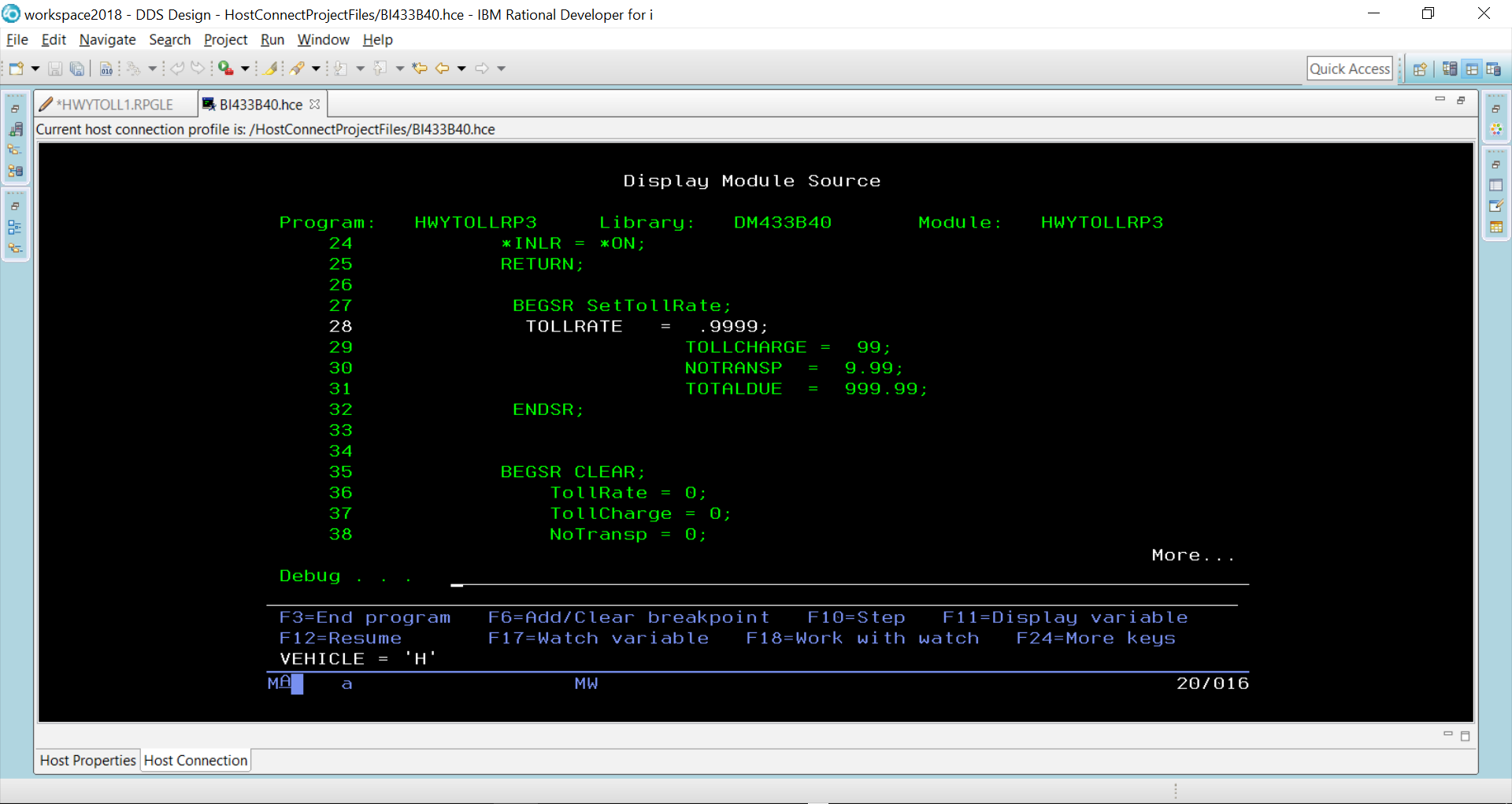
EVAL VEHICLE



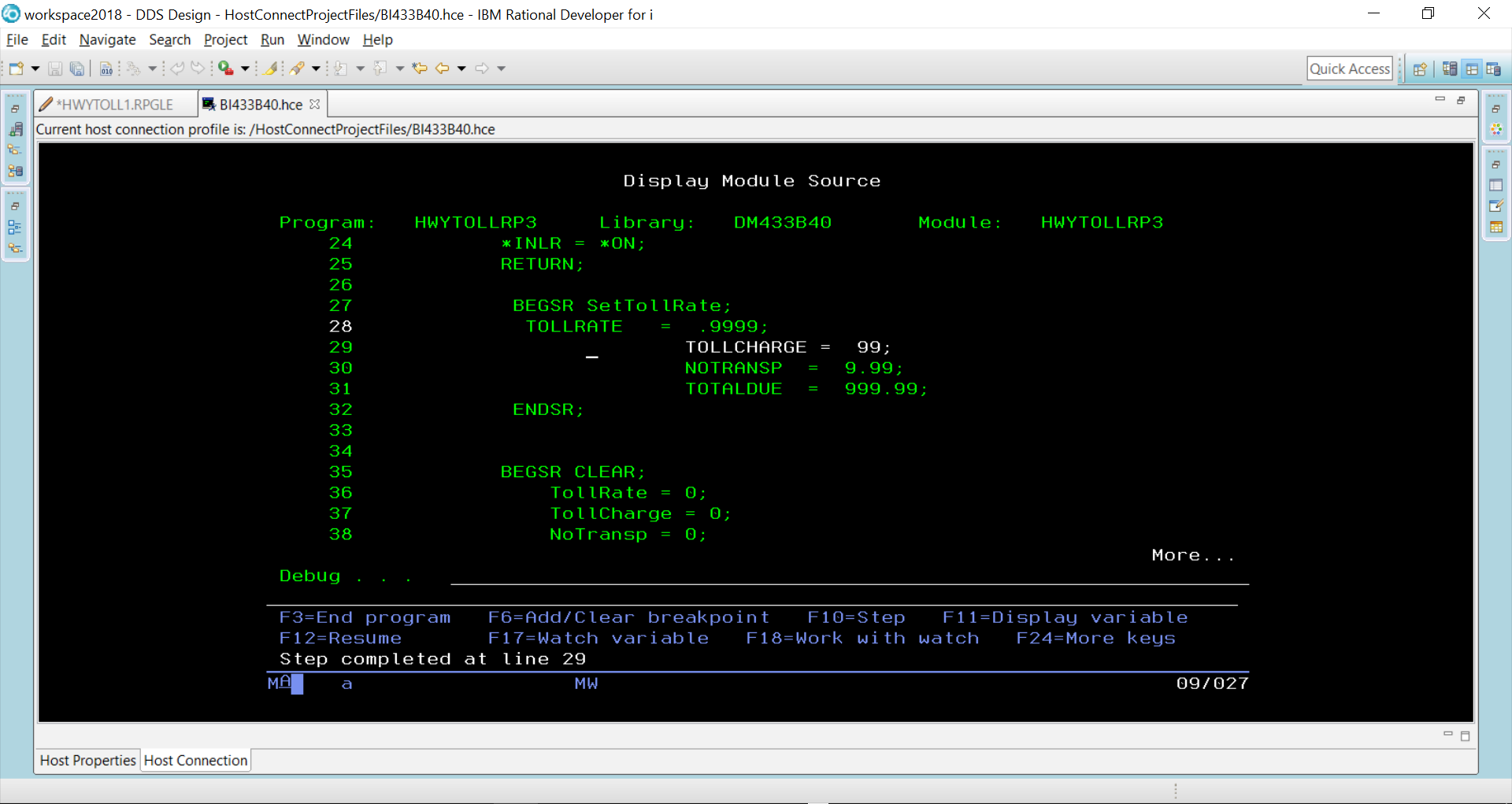
You can alter the contents of a field while your program is running.

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.

