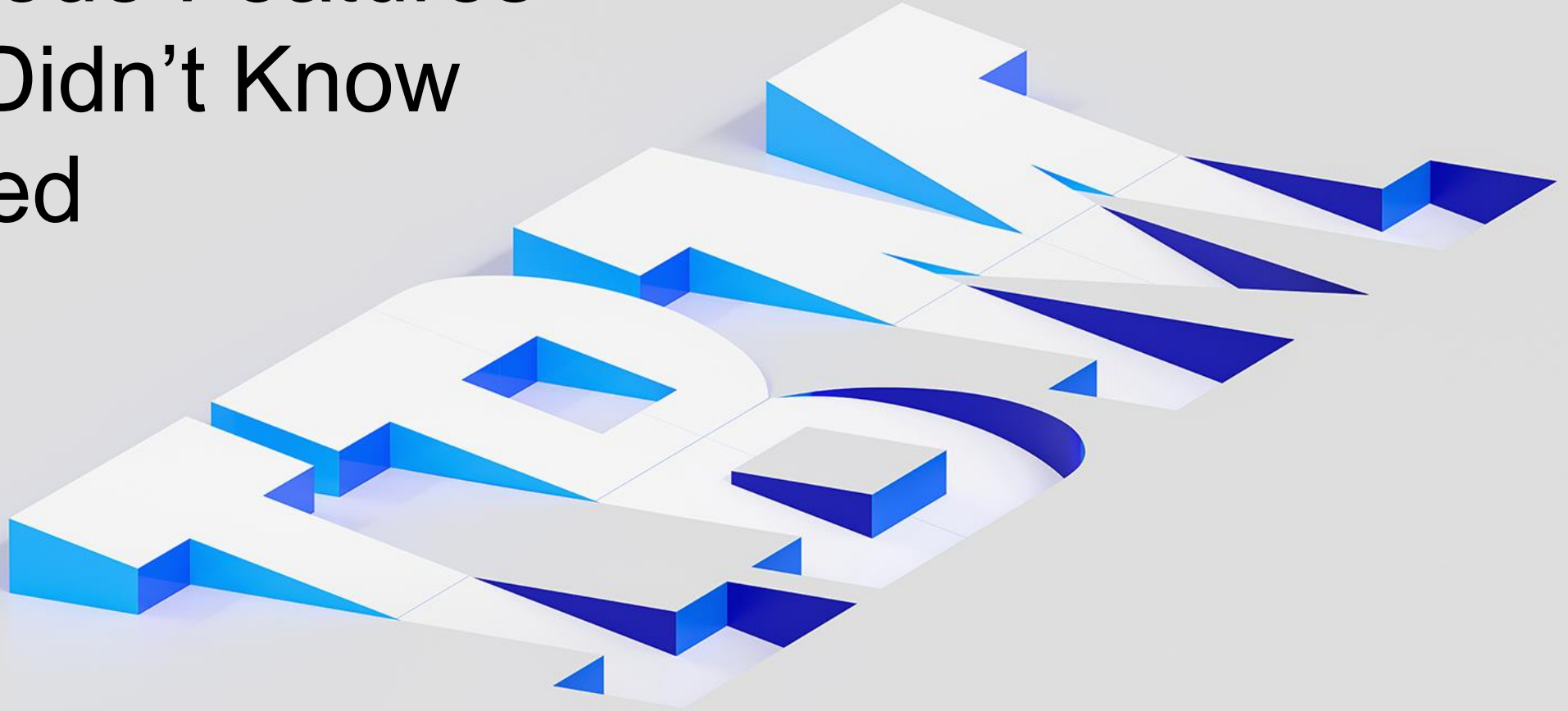
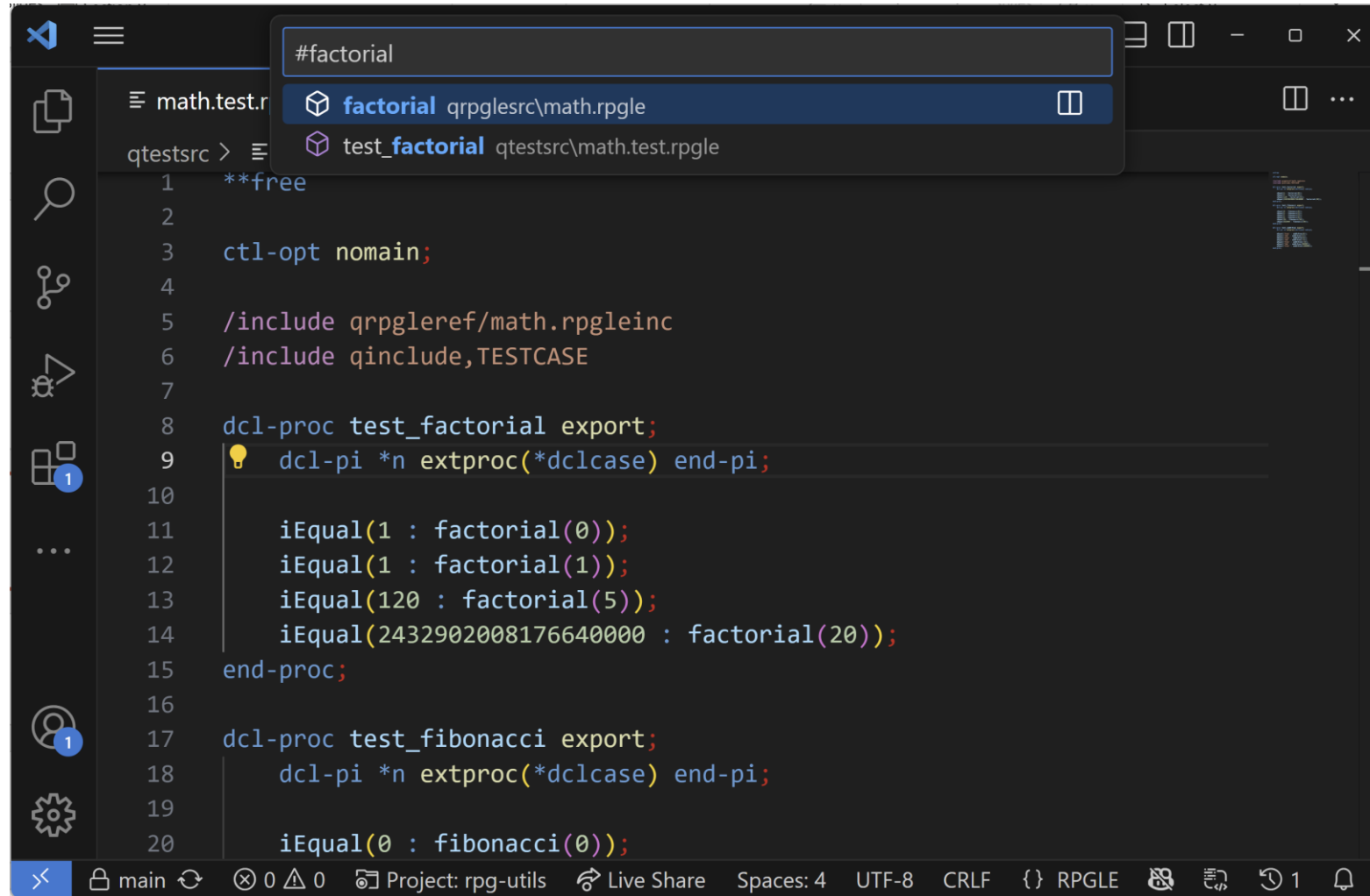


# VS Code Features You Didn't Know Existed



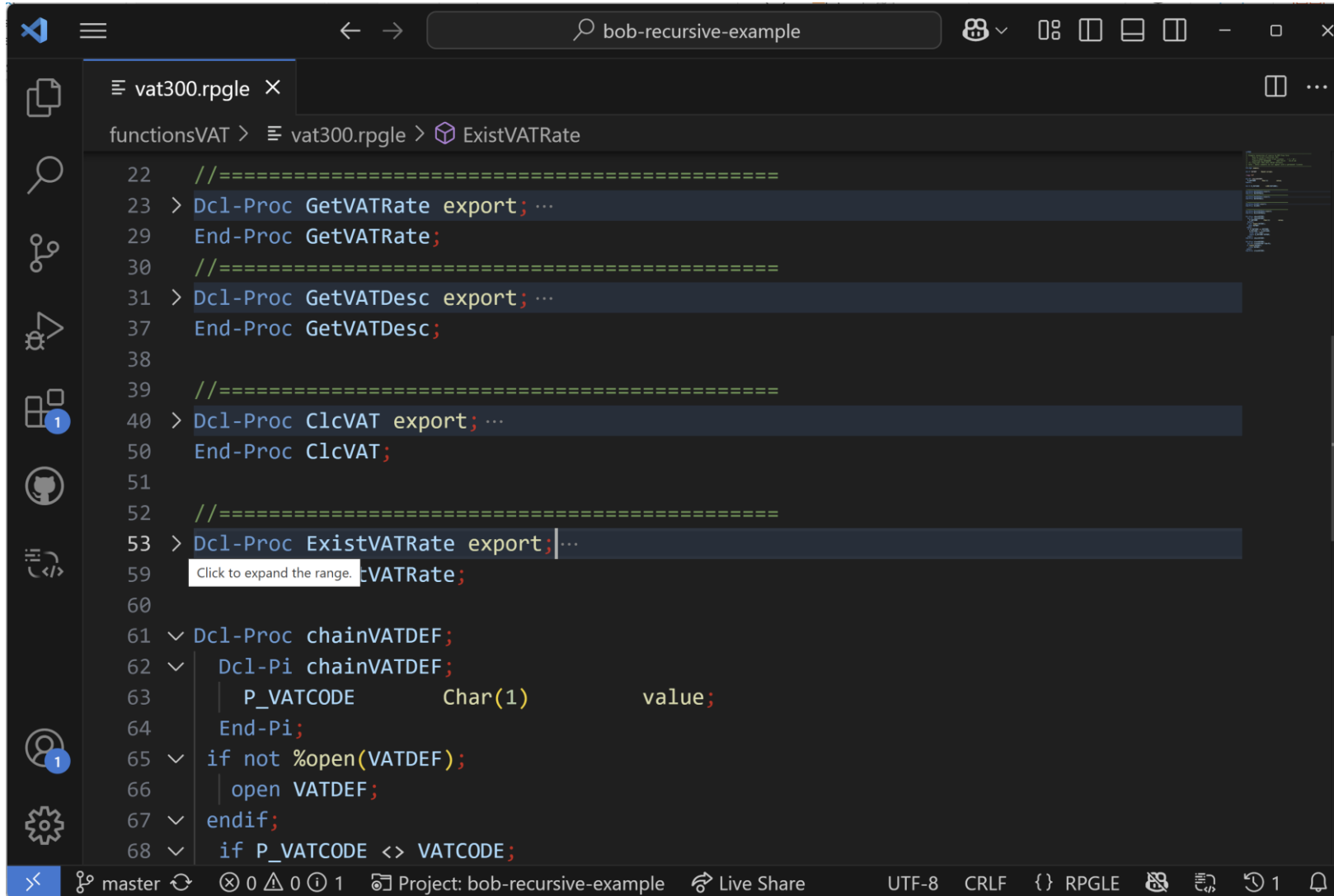
# Code Navigation – Go To Symbol in Editor (or Workspace)

Use the command “Go To Symbol in Editor”, “Go To Symbol in Editor by Category” or “Go To Symbol in Workspace”



# Code Navigation – Code Folding

Works on procedures, prototypes, data structures, etc.

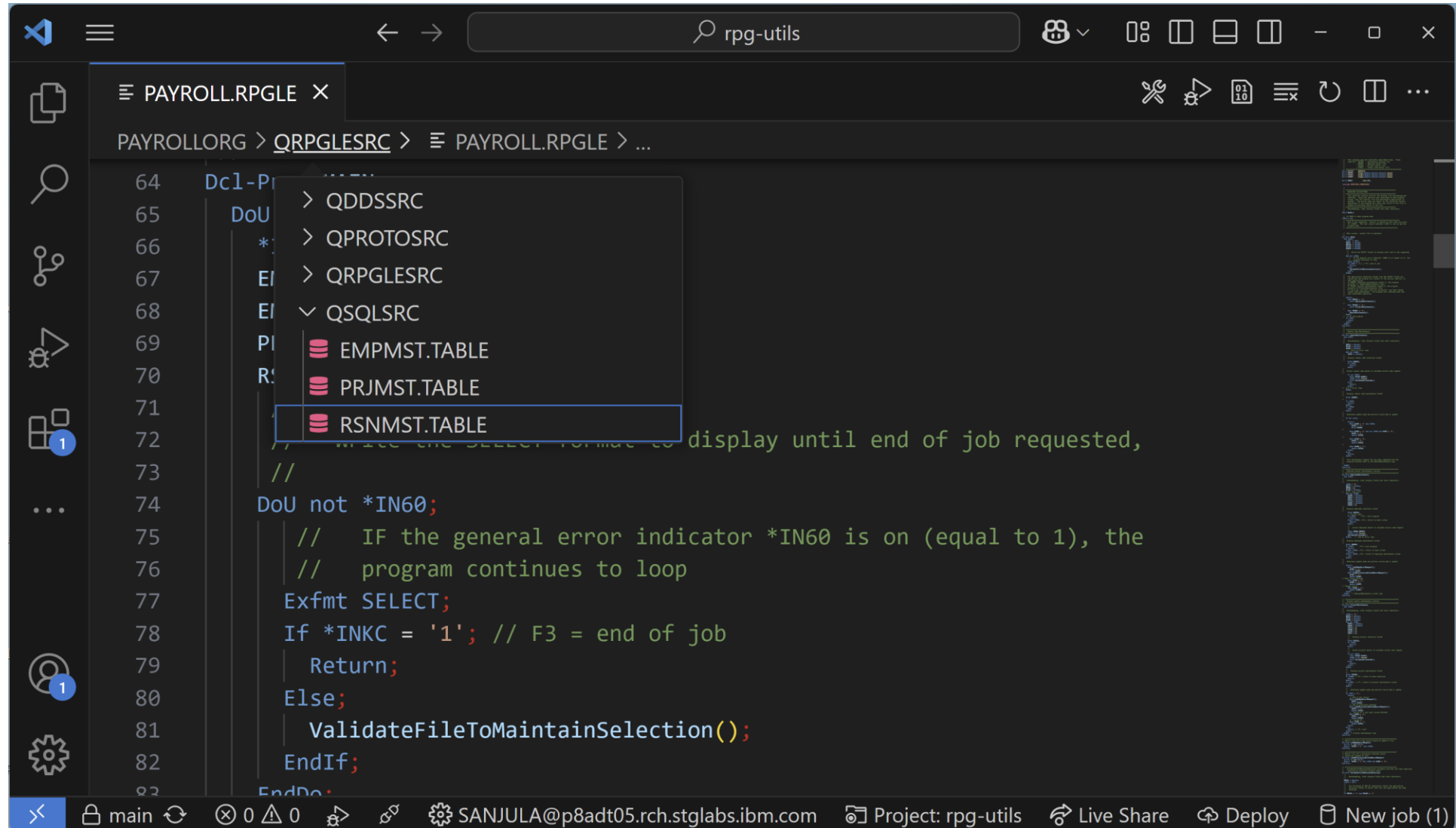


The screenshot shows the Visual Studio Code editor with a file named `vat300.rpgle` open. The breadcrumb navigation shows `functionsVAT > vat300.rpgle > ExistVATRate`. The code is written in RPGLE and includes several procedure declarations. The `ExistVATRate` procedure is currently expanded, showing its definition from line 53 to 68. Other procedures like `GetVATRate`, `GetVATDesc`, and `ClcVAT` are collapsed. The status bar at the bottom indicates the current branch is `master`, there are 0 errors and 1 warning, and the file is part of a project named `bob-recursive-example`.

```
22 //=====
23 > Dcl-Proc GetVATRate export; ...
29 End-Proc GetVATRate;
30 //=====
31 > Dcl-Proc GetVATDesc export; ...
37 End-Proc GetVATDesc;
38
39 //=====
40 > Dcl-Proc ClcVAT export; ...
50 End-Proc ClcVAT;
51
52 //=====
53 > Dcl-Proc ExistVATRate export; ...
59 Click to expand the range. tVATRate;
60
61 v Dcl-Proc chainVATDEF;
62 v Dcl-Pi chainVATDEF;
63 | P_VATCODE Char(1) value;
64 End-Pi;
65 v if not %open(VATDEF);
66 | open VATDEF;
67 v endif;
68 v if P_VATCODE <> VATCODE;
```

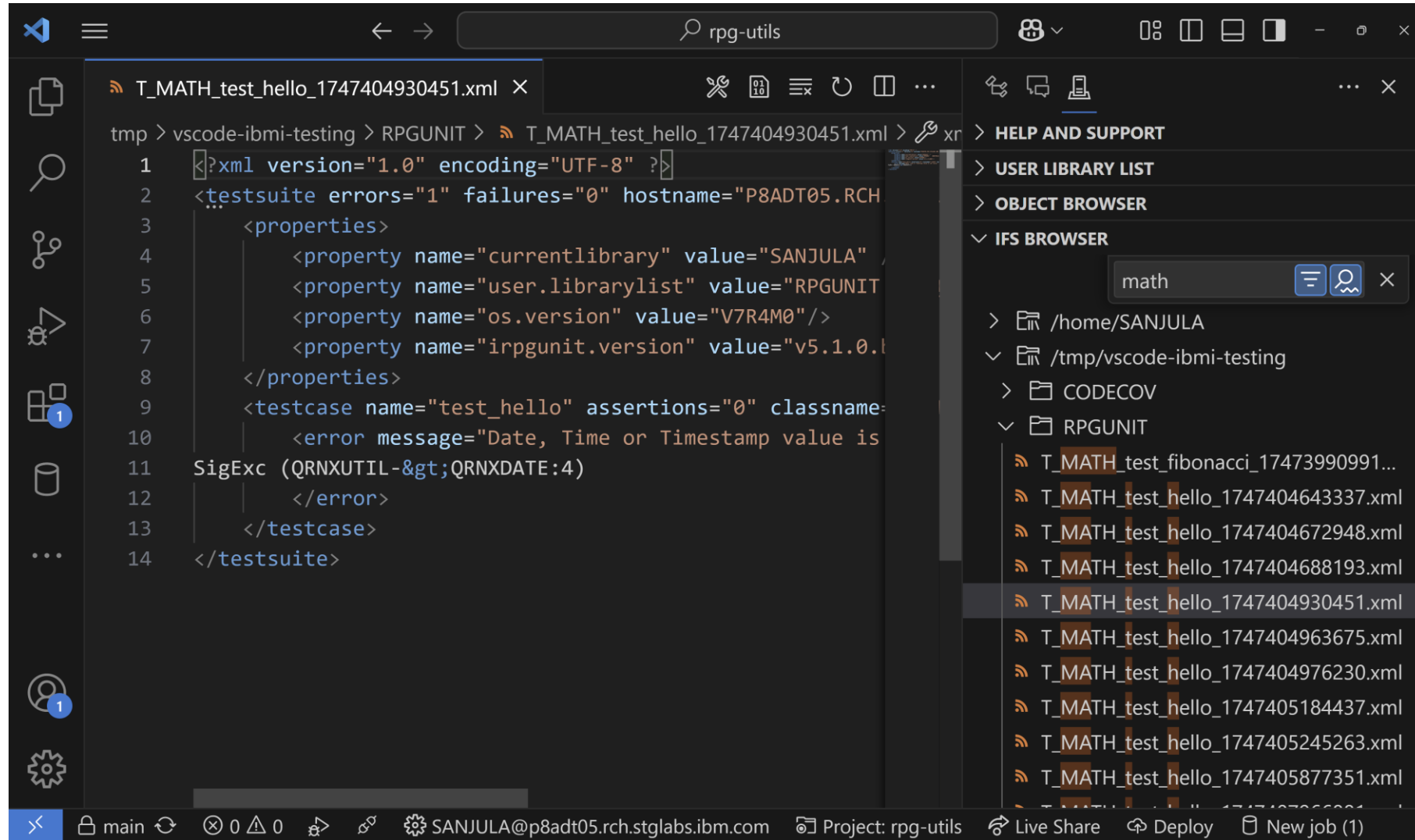
# Code Navigation – Breadcrumbs

Works in local, QSYS, and IFS file systems!



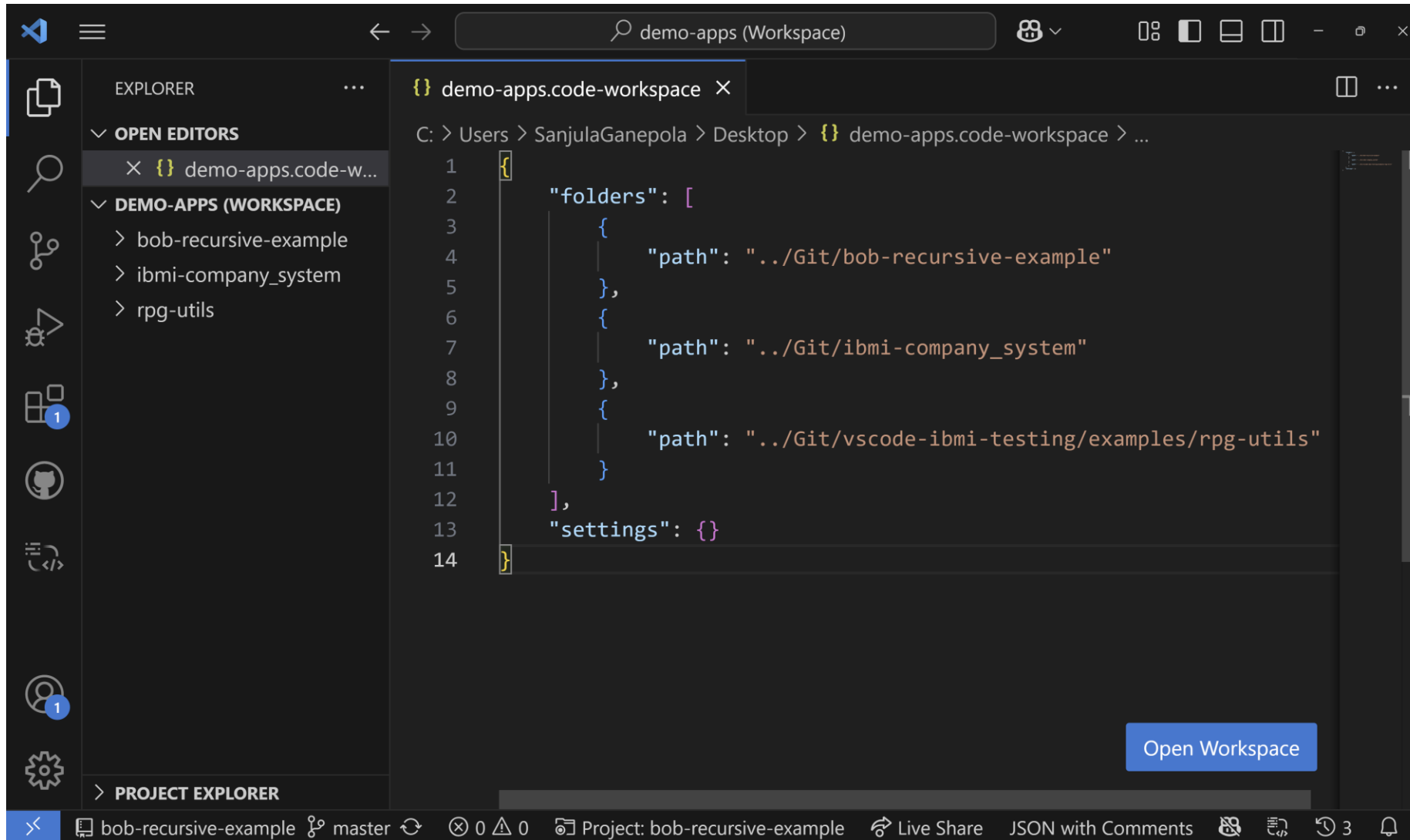
# Code Navigation – Tree View Searching

Use “Ctrl + Alt + f” in any view in VS Code (including the Object/IFS Browser!)



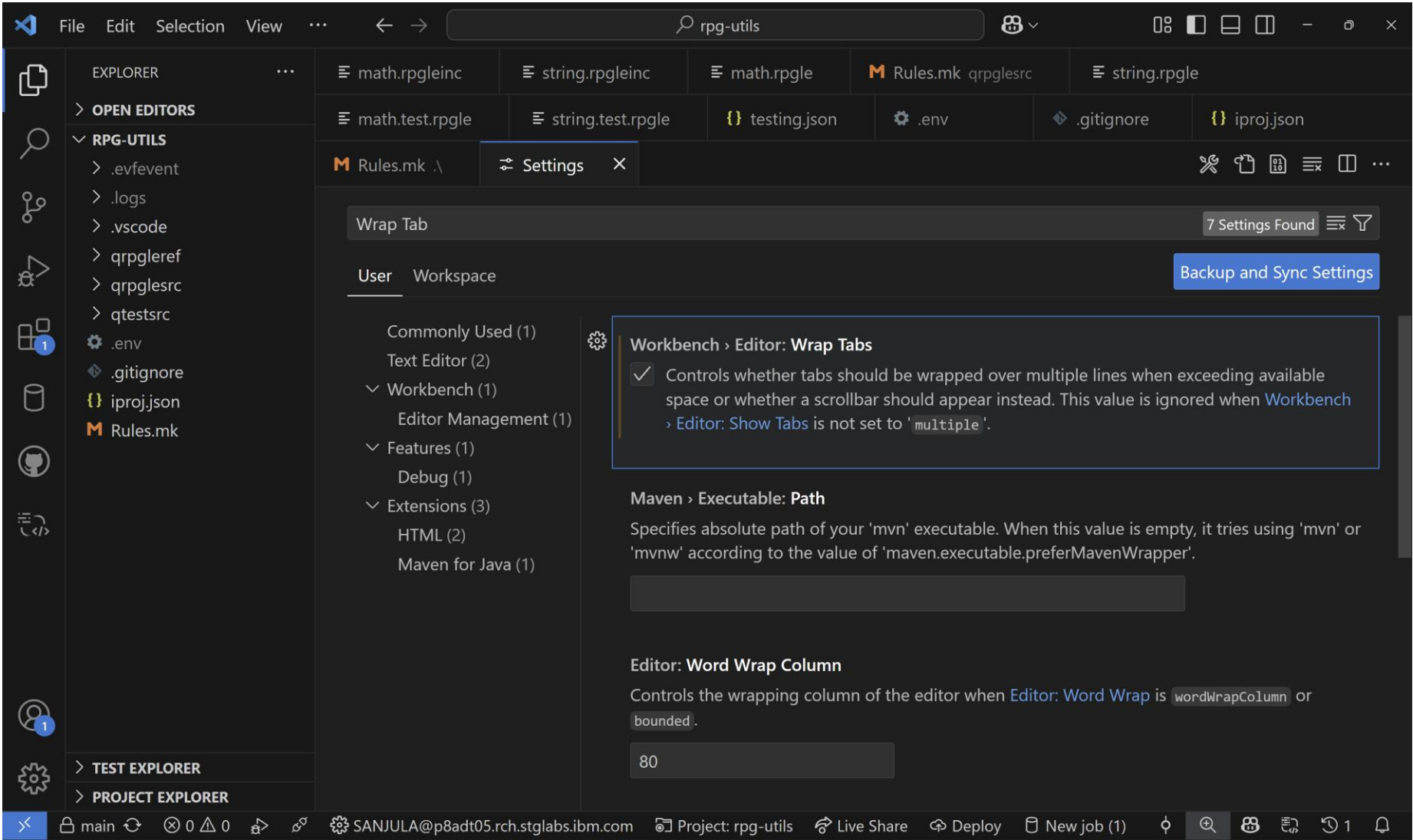
# Code Navigation – Multi-root Workspaces

Use the command “Add Folder to Workspace” and “Save Workspace As”



# Settings – Wrap Tabs

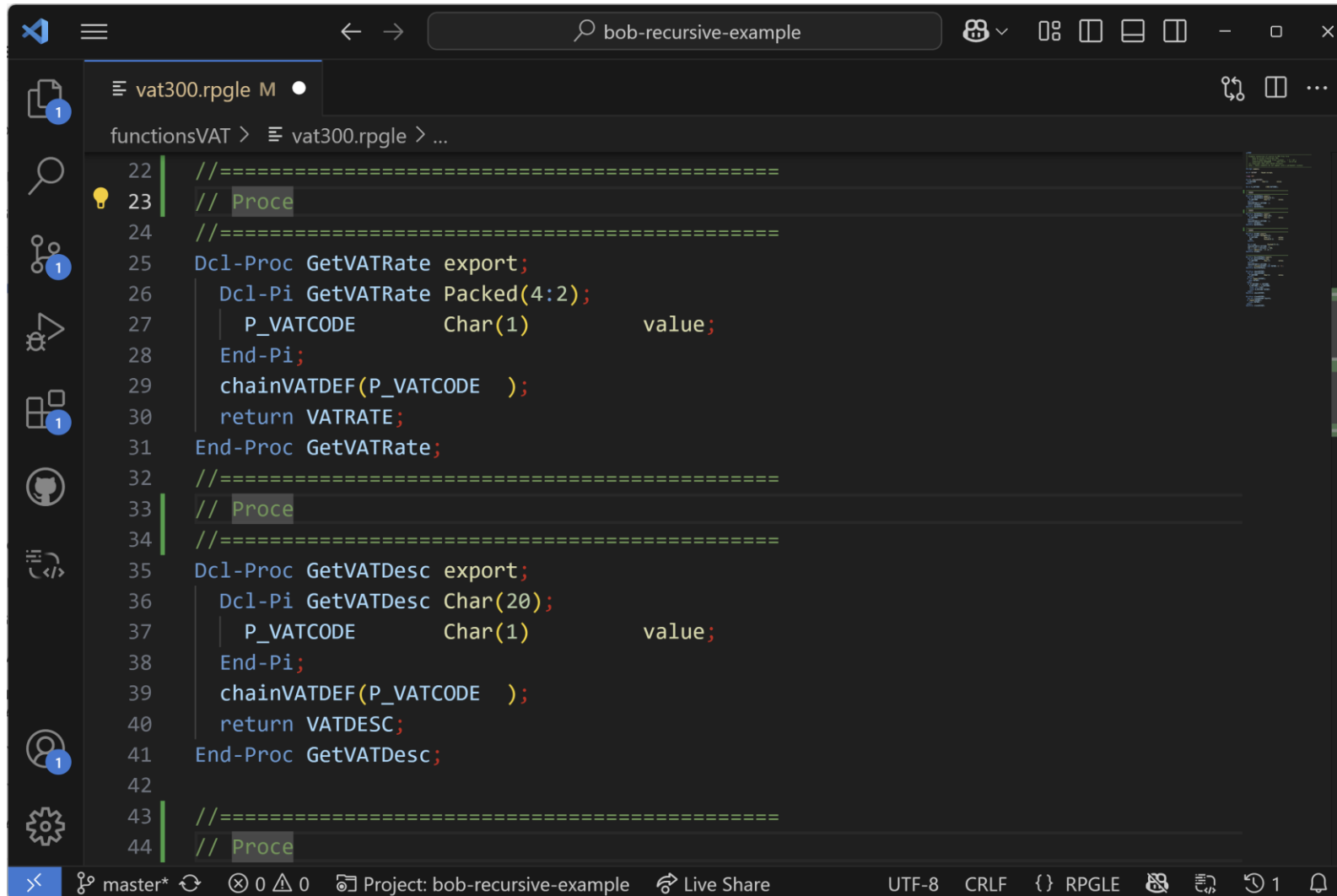
Enable the “Wrap Tabs” VS Code setting



# Code Editing – Multi-cursor

Manually add secondary cursors →  
Keyboard shortcut →

Alt + Click  
Ctrl + Alt + Down or Ctrl + Alt + Up



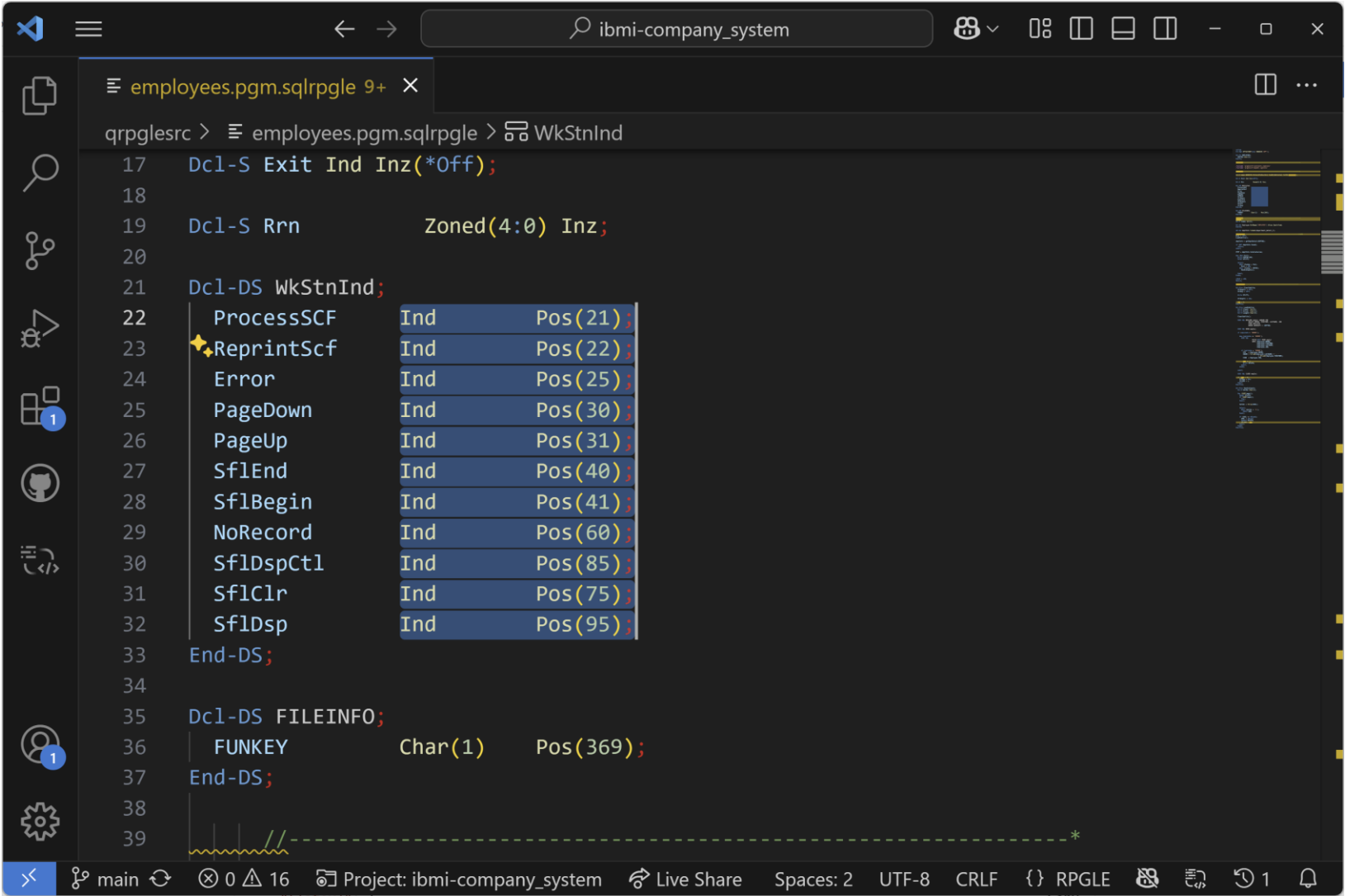
The screenshot shows the Visual Studio Code editor interface with a file named `vat300.rpgle` open. The editor displays RPGLE code with three secondary cursors (indicated by vertical green lines) placed on lines 23, 33, and 44, all of which start with `// Proce`. The code includes two procedure definitions: `GetVATRate` (lines 25-31) and `GetVATDesc` (lines 35-41). The status bar at the bottom shows the project name `bob-recursive-example`, encoding `UTF-8`, and line 1 of the file.

```
22 //=====
23 // Proce
24 //=====
25 Dcl-Proc GetVATRate export;
26   Dcl-Pi GetVATRate Packed(4:2);
27     P_VATCODE      Char(1)      value;
28   End-Pi;
29   chainVATDEF(P_VATCODE );
30   return VATRATE;
31 End-Proc GetVATRate;
32 //=====
33 // Proce
34 //=====
35 Dcl-Proc GetVATDesc export;
36   Dcl-Pi GetVATDesc Char(20);
37     P_VATCODE      Char(1)      value;
38   End-Pi;
39   chainVATDEF(P_VATCODE );
40   return VATDESC;
41 End-Proc GetVATDesc;
42
43 //=====
44 // Proce
```



# Code Editing – Column (box) selection

Shift + Alt while dragging



The screenshot shows the Visual Studio Code editor with a file named `employees.pgm.sqlrpgle` open. The editor is displaying a list of data structures (DS) for `WkStnInd`. The code is as follows:

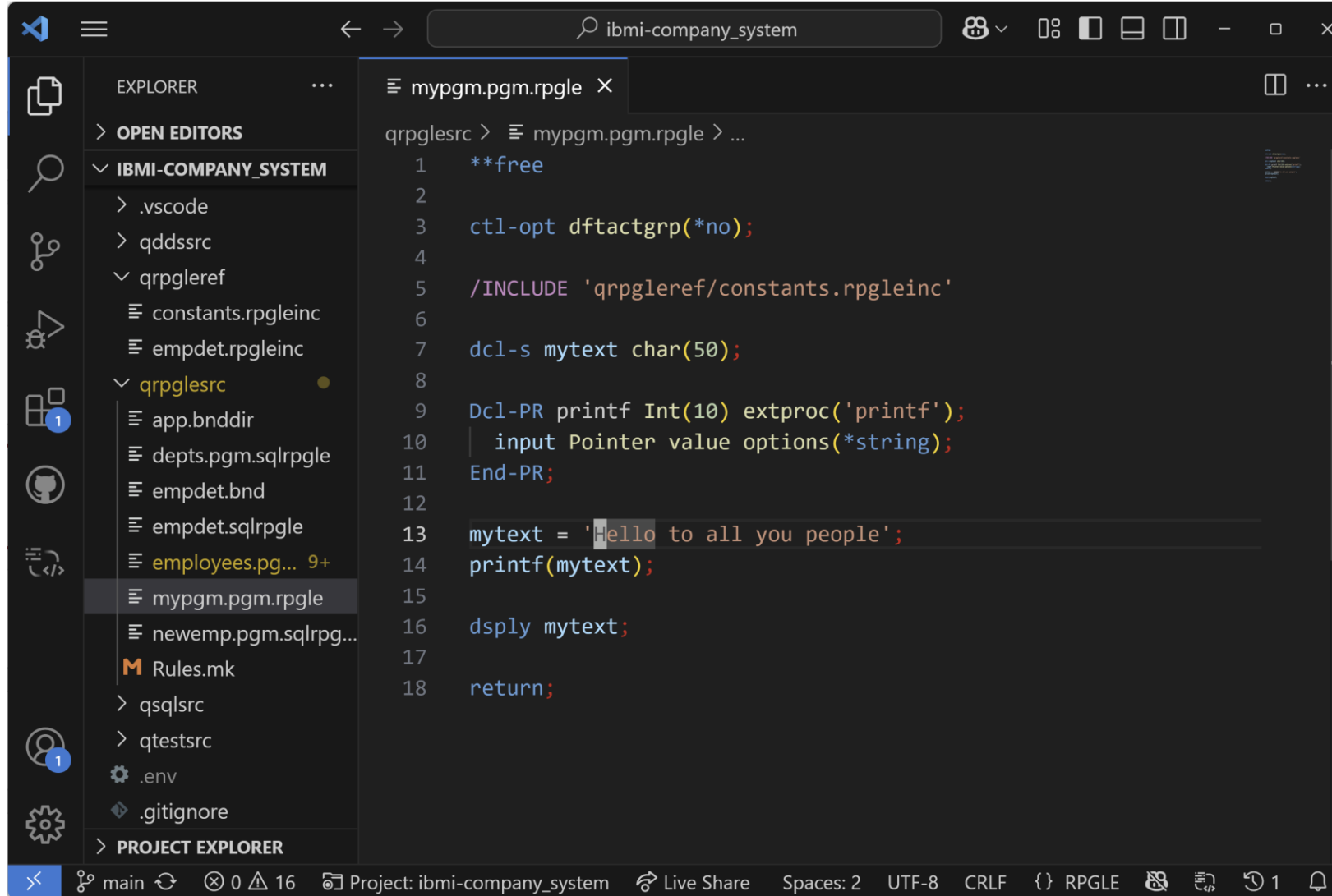
```
17 Dcl-S Exit Ind Inz(*Off);
18
19 Dcl-S Rrn          Zoned(4:0) Inz;
20
21 Dcl-DS WkStnInd;
22   ProcessSCF      Ind      Pos(21);
23   *ReprintScf     Ind      Pos(22);
24   Error           Ind      Pos(25);
25   PageDown        Ind      Pos(30);
26   PageUp          Ind      Pos(31);
27   SflEnd          Ind      Pos(40);
28   SflBegin        Ind      Pos(41);
29   NoRecord        Ind      Pos(60);
30   SflDspCtl       Ind      Pos(85);
31   SflClr          Ind      Pos(75);
32   SflDsp          Ind      Pos(95);
33 End-DS;
34
35 Dcl-DS FILEINFO;
36   FUNKEY          Char(1)   Pos(369);
37 End-DS;
38
39
```

The code is being edited in a dark theme. The editor shows a list of data structures (DS) for `WkStnInd`. The code is as follows:

The status bar at the bottom indicates the project is `ibmi-company_system`, the file is `main`, and the encoding is `UTF-8`. The status bar also shows `Spaces: 2`, `CRLF`, and `{ } RPGLE`.

# Code Editing – Overtyp vs. Insert Mode

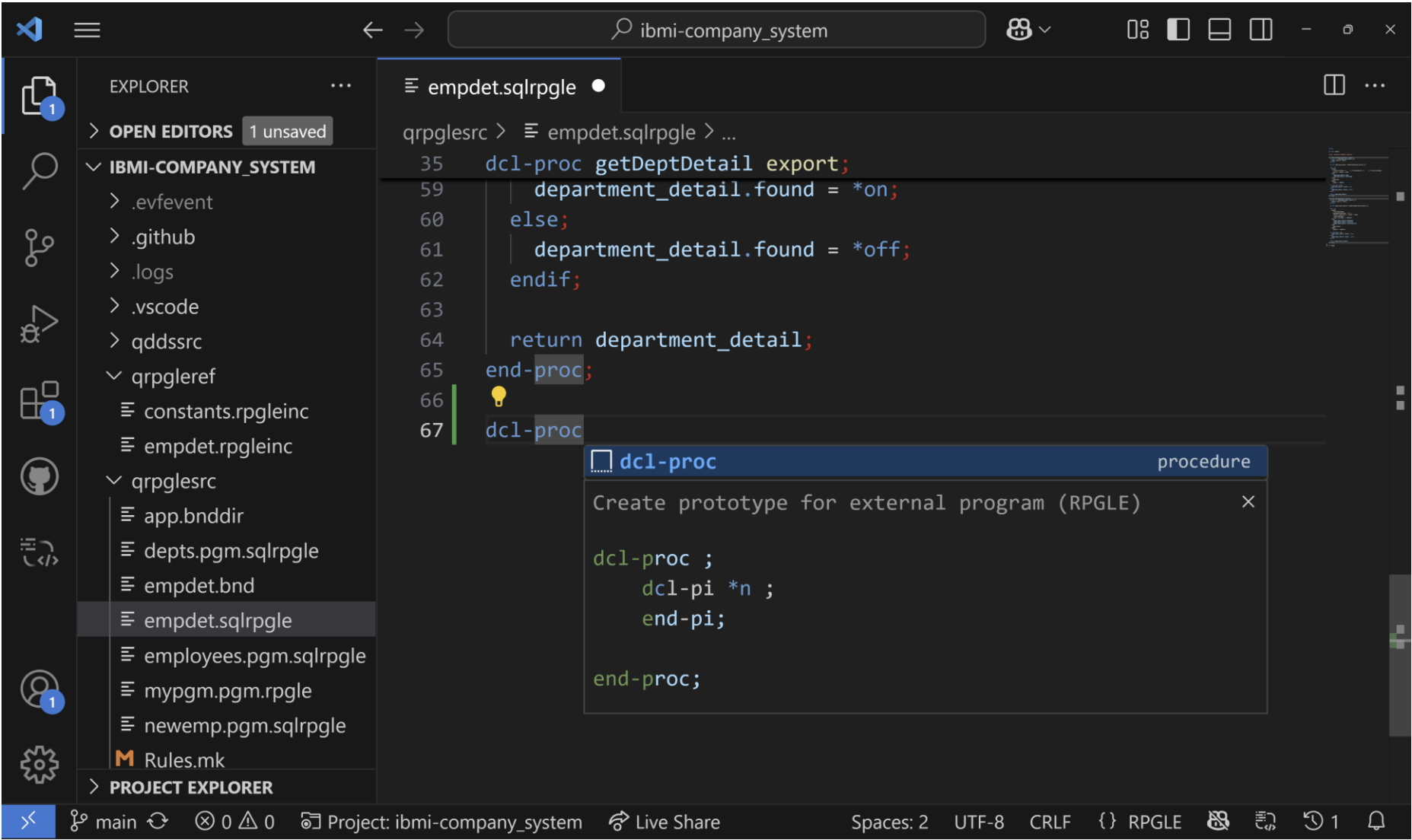
Use the command “Toggle Overtyp/Insert Mode”



```
qrpglesrc > mypgm.pgm.rpgle > ...
1  **free
2
3  ctl-opt dftactgrp(*no);
4
5  /INCLUDE 'qrpgleref/constants.rpgleinc'
6
7  dcl-s mytext char(50);
8
9  Dcl-PR printf Int(10) extproc('printf');
10 |   input Pointer value options(*string);
11 End-PR;
12
13 mytext = 'Hello to all you people';
14 printf(mytext);
15
16 dsply mytext;
17
18 return;
```

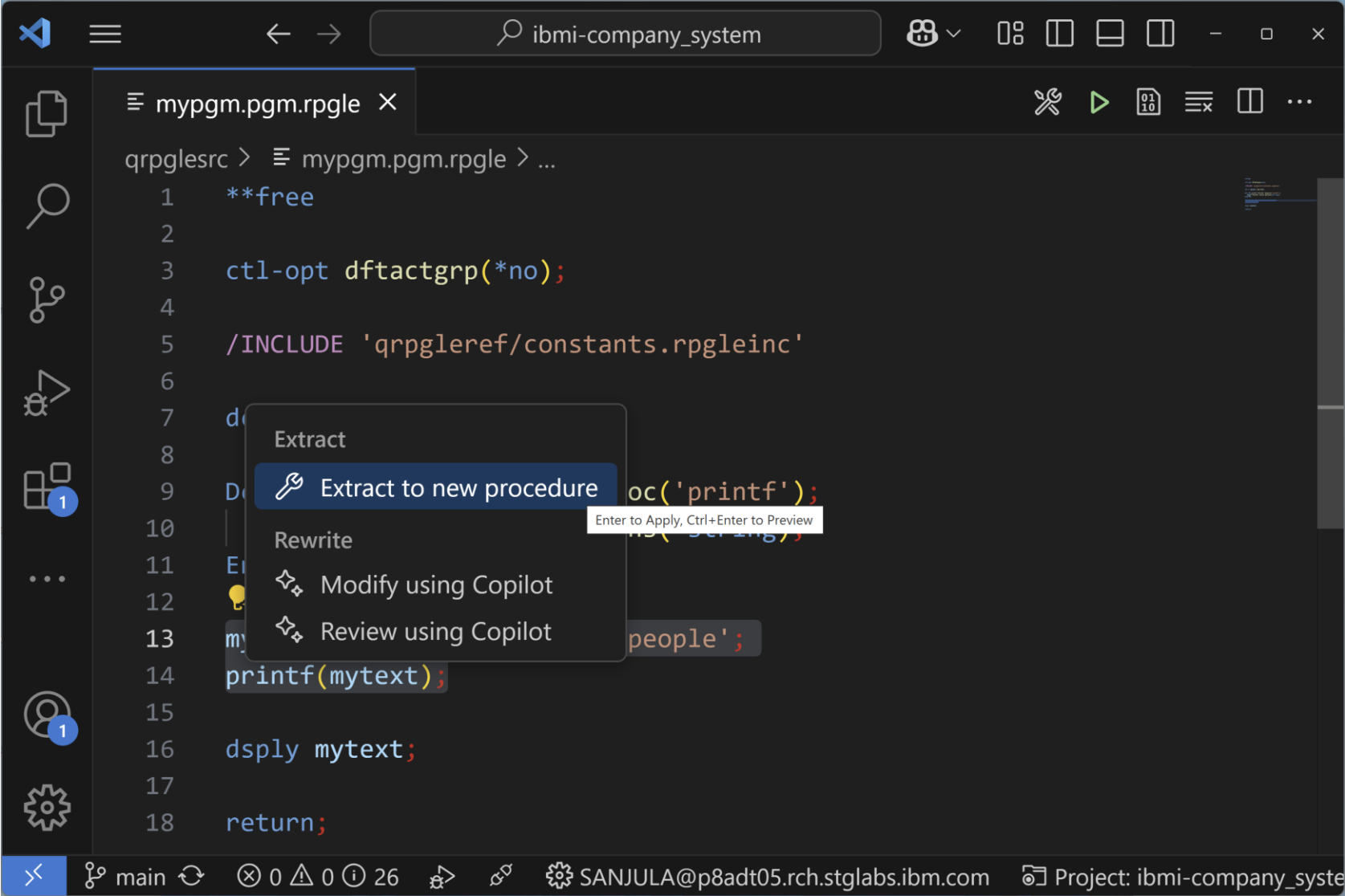
# Code Editing – Code Snippets

Start typing the prefix for a snippet or use “Insert Snippet”



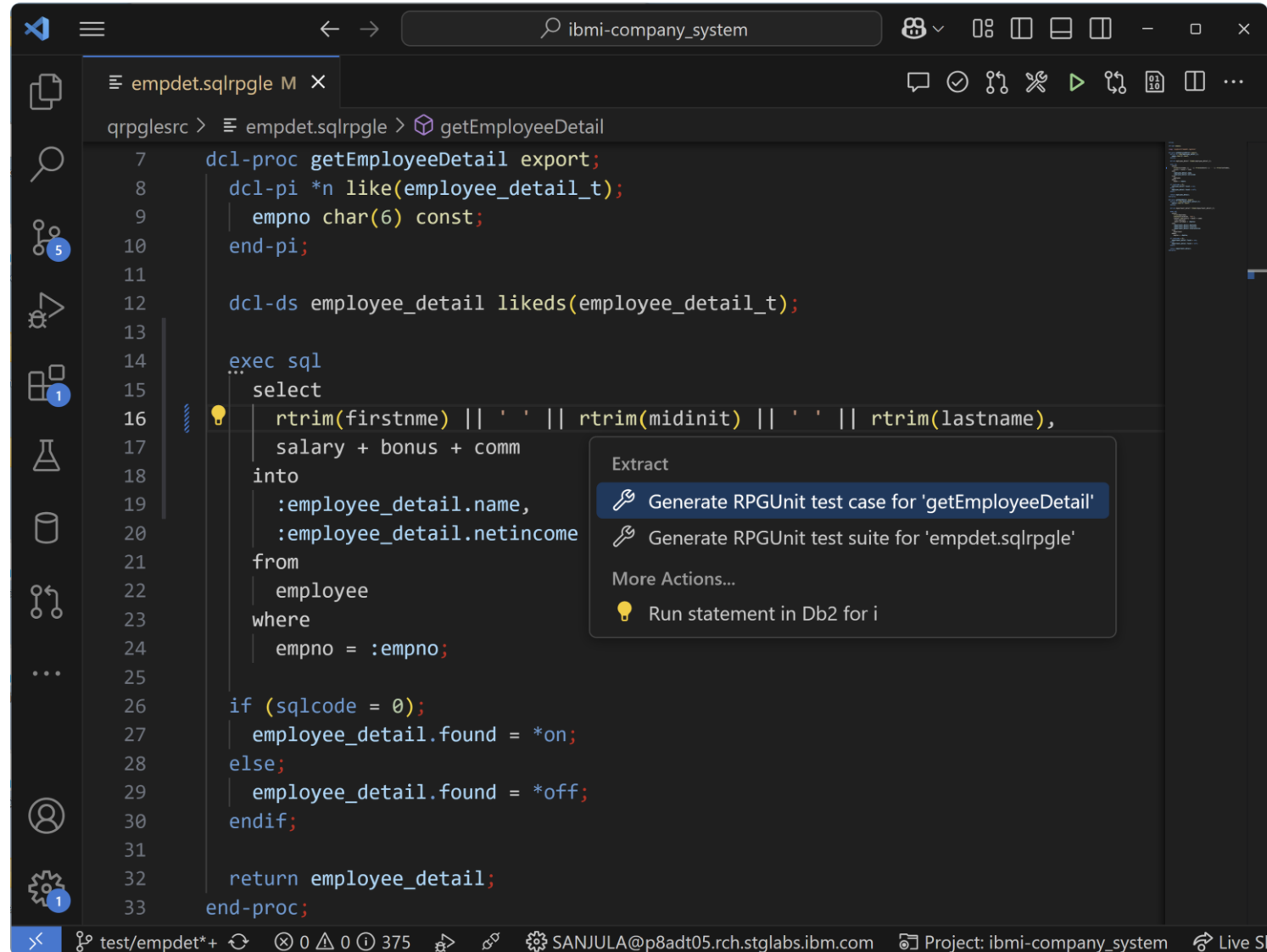
# Code Editing – Code Actions (Extract to new procedure)

Extract code into a procedure



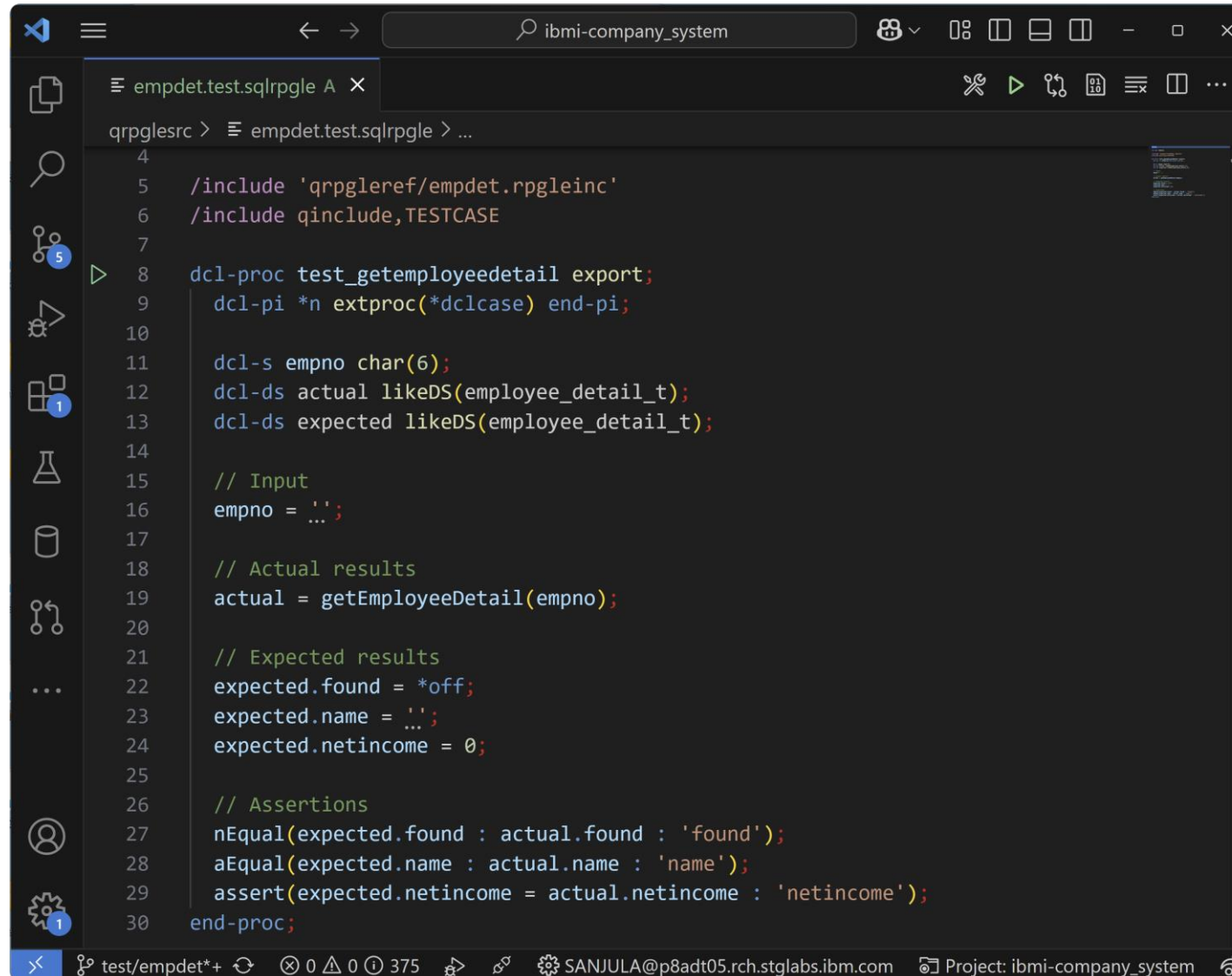
# Code Editing – Code Actions (Test Stub Generation)

Generate RPGUnit test case or test suite



# Code Editing – Fill in Test Stub

Fill in the test stub with inputs and expected results

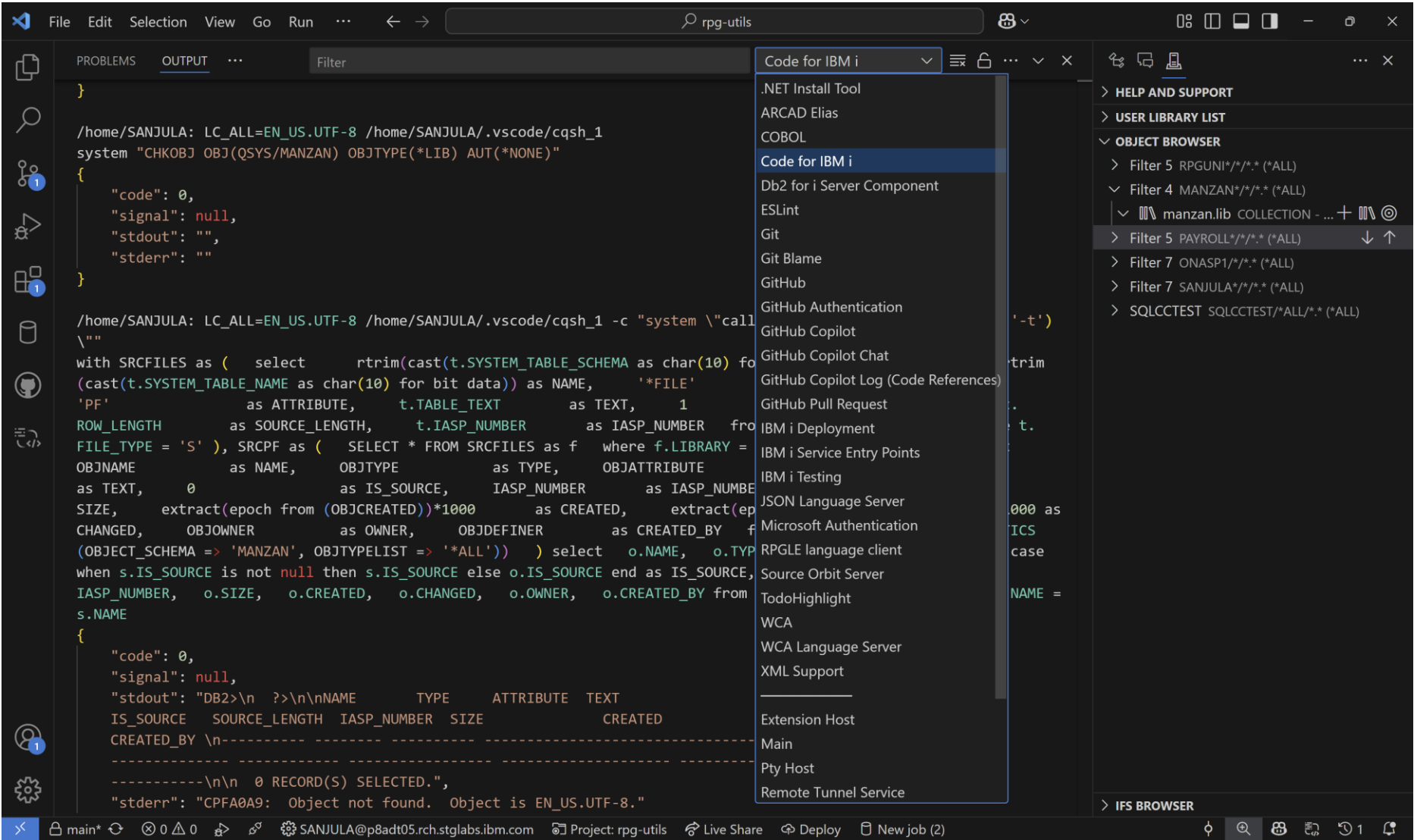


The screenshot shows a code editor window with a dark theme. The title bar indicates the project is 'ibmi-company\_system'. The editor displays a COBOL test stub file named 'empdet.test.sqlrpgle'. The code includes include statements for 'qrpglef/empdet.rpgleinc' and 'qinclude,TESTCASE'. It defines a procedure 'test\_getemployeedetail' with an 'export' attribute. Inside the procedure, it declares a variable 'empno' of type 'char(6)', and two data structures 'actual' and 'expected' of type 'likeDS(employee\_detail\_t)'. The code then sets 'empno' to an empty string, calls 'getEmployeeDetail' to populate 'actual', and initializes 'expected' with 'found' set to 'off', 'name' to an empty string, and 'netincome' to 0. Finally, it performs assertions: 'nEqual' for 'found', 'aEqual' for 'name', and 'assert' for 'netincome'. The procedure ends with 'end-proc;'. The status bar at the bottom shows the current file is 'test/empdet\*+', there are 0 errors and 0 warnings, 375 lines of code, and the user is 'SANJULA@p8adt05.rch.stglabs.ibm.com' working on the 'Project: ibmi-company\_system'.

```
4  
5  /include 'qrpglef/empdet.rpgleinc'  
6  /include qinclude,TESTCASE  
7  
8  dcl-proc test_getemployeedetail export;  
9    dcl-pi *n extproc(*dclcase) end-pi;  
10  
11    dcl-s empno char(6);  
12    dcl-ds actual likeDS(employee_detail_t);  
13    dcl-ds expected likeDS(employee_detail_t);  
14  
15    // Input  
16    empno = '';  
17  
18    // Actual results  
19    actual = getEmployeeDetail(empno);  
20  
21    // Expected results  
22    expected.found = *off;  
23    expected.name = '';  
24    expected.netincome = 0;  
25  
26    // Assertions  
27    nEqual(expected.found : actual.found : 'found');  
28    aEqual(expected.name : actual.name : 'name');  
29    assert(expected.netincome = actual.netincome : 'netincome');  
30  end-proc;
```

# Extensions – Output Channels

Diagnose extension issues using the extension's output channel



# Sanjula Ganepola - VS Code Features You Didn't Know Existed

Please take the last minute of this session to complete the evaluation. A direct link to the evaluation can be found using the QR code to the right.





**IBM i**