5321 Homework 5

Fall 2019

Question Weighting:

Question 1-5 - 20% each

Use the following approach for each problem **(all test must use the JUnitParams runner and read values from csv files).**

1. Develop CFG (reduced) and cyclomatic complexity.
2. Develop basis path set.
3. Determine significance on each variable.
4. Develop inputs and expected outputs from requirements, not code.
5. Add tests for missing Boundary Values not tested, including extreme range values.
6. Add tests for extreme range values for each variable that has a BV.
7. Add MCDC test cases for Multiple Condition Decision statements.
8. Add test cases to verify all table data.

**Submit the following in the PDF file - this is the evidence file**

1. Test case table snapshot
   1. Basis Path test cases (for problem 1-2 only)
      1. Use the line numbers in Eclipse for your basis path line.
      2. where tests are addition to basis path set use a "-" to indicate the basis path.
      3. Make sure all true is the first BP and tests are in correct order
      4. You do NOT need to submit the CFG with this homework
   2. Indicate tests for MCDC coverage with a "Statement 11 FFT" - put this in the comment column.
2. JUnit pass indicator (green bar expanded)
3. JaCoCo statement green source line annotations (not summary)
4. Make sure to include the time stamp on your screen shots.

**Include in ZIP file**

1. Your evidence file (JaCoCo/JUnit screenshot)
2. JUnit test files (make sure problem number is referenced in the file name)
3. csv files used (make sure problem number is referenced in the file name)
4. For Problem 4 your Problem4ServerData.class file
5. For problem 5
   1. both PIT html files
   2. Text explanation of PIT coverage analysis of the provided tests

**Problem 1**) Test the Problem 1 source code (in the attached zip file).

**Assumptions**

1. cart ranges from $0.00 to $20,000.00 both inclusive.
2. For result use Excel's currency format and do not truncate.
3. memberPoints ranges from 0 to 10,000 both inclusive
4. items ranges from 1 to 50 both inclusive
5. use taxRate = 8.25%

**Test**

Use the following template for the test case table. Document how the multiple condition expression is tested using MCDC in the Comments column (see the Test case table below) and as described above. The comments column should also describe the basis path set.



Use the same inputs as shown above **only changing the value of cart** for the following test cases:

1. Basis path set
2. Missing BVs from the Basis path set
3. Extreme range values (for memberPoints and items extreme ranges use cart value from above).

Test case design

1. The comparison threshold for result is 0.006 in JUnit
2. You must use VLOOKUP to compute the value of result in your table. Please see the test case table for problem 2 from HW 3 to see how to use this function. Note that the function references another tab that has the BVs for each ECP with the corresponding values of discount for each.

**Problem 2**) Test the Problem 2 source code (in the attached zip file).

**Assumptions**

1. distance ranges from 0.0 to 5,000.0 ft both inclusive. Significance of 0.1
2. speed ranges from 0.0 to 150.0 mph both inclusive. Significance of 0.1
3. For brakingFactor use Excel's number format and do not truncate. Significance of 0.01

**Test**

Use the following template for the test case table. Document how the multiple condition expression is tested using MCDC in the Comments column (see the Test case table below) and as described above. The comments column should also describe the basis path set.



Use the same inputs as shown above **only changing the value of distance** for the following test cases:

1. Basis path set
2. Missing BVs from the Basis path set
3. Extreme range values (for speed extreme range values use the value of distance shown above).

Test case design

1. The comparison threshold for brakingFactor is 0.001 in JUnit
2. You must use VLOOKUP to compute the value of result in your table. Please see the test case table for problem 2 from HW 3 to see how to use this function. Note that the function references another tab that has the BVs for each ECP with the corresponding values of distance for each.

**Problem 3**) Test the Problem 3 source code (in the attached zip file). **Do not use Basis path for this problem.**

**Description**

This code provides the absolute day number of the year for the preceding day.

1. Jan 02, 2019 would be 1
2. Dec 31, 2019 would be 364
3. Jan 01, 2020 would be 365
4. Jan 01, 2021 would be 366

It accounts for leap years.

**Assumptions**

1. range for year is 2019-2402 both inclusive
2. range for month is 1 to 12 both inclusive
3. range for day is 1 to 31 both inclusive
4. **DO NOT** perform extreme range tests on day, month, and/or year

**Test**

Use the following template for the test case table. Document how the multiple condition expression is tested using MCDC in the Comments column (see the Test case table below) - note that statements 8,9, and 12 need to be described in this column.



Test case design. In addition to testing for basis path, BVs, MCDC on multiple condition statements, your tests must test all table data. **Use the smallest year that fits the test case you are trying to use.**

**Problem 4**) Test the Problem 4 source code (in the attached zip file).

Use the supplied Excel test case table (which is the solution for problem 1 from HW 3).

Use EasyMock to mock the call to the server to get batteryPower.

EasyMock instructions

* + 1. Download EasyMock from the M14 Blackboard files
    2. InstallEasyMock in your project (add the EasyMock.jar to your Java Build path)

Execute the JUnit test. Create **Problem4ServerData.java** to define the signature for the getBatteryPower() method (see slide 32 of M14 for how to do this). Follow the five steps shown in slide 35 of M14 to get EasyMock to work in your test environment.

**Problem 5**) Test the Problem 5 source code (in the attached zip file) which is actually problem 1 from this assignment. Use the test case table provided to test the file. Run PIT against this test (use v1.1.9 - see the last line of the index.html file). Run JUnit and JaCoCo coverage against the tests. Take snapshots of each as required.

**Determine what is wrong with the PIT coverage**. Explain in your own words why the PIT coverage is so poor. Compare these test cases with problem 1 tests and take a PIT snapshot of each (both Problem 1 and this problem's test results).

**Submit**

1. Text explanation of PIT coverage analysis of the provided tests (considering the JUnit and JaCoCo coverage indicated).
2. JUnit and JaCoCo coverage snapshot with timestamp
3. Two PIT html files (one each for Problem 1 and Problem 5 PIT results). Place these two files in your zip file in a folder named **Problem5 PIT results**.

To get the html files look under your Eclipse Workspace -> .metadata -> .plugins -> org.pitest.pitclipse.core -> html\_results ... and find files: Problem1Class.java.html and Problem5Class.java.html.

**Double check**

1. Make sure that you have PIT v1.1.9 (see the last line in the PIT html results file).
2. Make sure that all Mutators is set. 50% deduction if not.