Project Assignment 2:
Specification-Based and Control-Flow-Based Testing

Due: Friday, 11/04/2022, 23:59h

This project is intended to prepare you for the exam and to give you some more practical experience in software quality assurance. Please work together with your homework partner and another homework team.

Task 1

Find the file you created for Project 1 containing the description of your project. Append the solutions for this project into the same file.

Task 2

Correct all the mistakes and incorporate all the feedback you received on Project Assignment 1.

Task 3

Create a set of **data-flow annotated control-flow graphs** for the entire that contains your selected Component (i.e. for all methods therein). The control-flow graphs should reflect the advancement of control flow through each method.

You may want to add a new Section 4 to your document (i.e. *after* your Section 3 on equivalence classes and *before* the *current* Section 4 on quality assurance mechanisms) called "Data-Flow Annotated Control Flow Graphs." Each subsection (i.e. 4.1, 4.2, 4.3, …) should correspond to one method in your component. To increase readability, be sure to maintain the same order of subsections as you did in Section 3; i.e., if Section 3.1 contained equivalence classes for method `boolean foo(int x)`, Section 4.1 should be the control flow graph for method `boolean foo(int x)`.

Task 4

Based on the control flow graphs for your methods from Task 2, derive test cases to achieve **complete coverage** for the following test techniques:

a) Statement Test

b) Decision/Branch Test

c) Minimal Multiple Condition Test

d) Path Coverage Test

e) Loop Test.

Be smart about this. Make clever use of things that subsume one another. If your method doesn't have loops, you can't test them, but you could test path coverage. But be sure to cover all paths, not "just" branches. Creating one or more truth tables to find test cases for b) and c) will help you find paths.

For each test case, document a unique ID, values for each parameter, expected outputs, covered path through the control flow graph. Also, annotate which for which test technique the test case is applicable.

Again, be clever about this – which test cases satisfy more than one criterion? Which test cases from Project 1 are applicable? Don't derive new test cases unless you have to, but document it, if you reuse test cases. It may be the case that for some criterion, complete coverage not achievable. If so, document your argument what it isn't.

You may want to add a *new* Section 5 (*after* your Section 4: Control Flow Graphs) called "Control Flow Based Tests." Work by analogy from Task 3 and create a suitable substructure. You may reuse the templates from the assignment sheets.

Task 5

Check, which test cases from Task 4 **satisfy the all-uses-criterion**. Do the following:

a) If you haven't done so already, add data-flow attributes to the control flow graphs from Task 3. Be sure to condense blocks of statements into single nodes and remember that loops, conditions, etc. are individual nodes as well. For each node, annotate the all defs, c-uses, and p-uses. (There's no need to include TWO sets of CFGs in your document. Simply include the data-flow annotated CFGs, like Task 2 says).

b) Determine the DEF set and the C-USE set for each node the P-USE sets for each branch. Remember that loops always include a condition! Document your findings using the appropriate templates, like we have seen in the assignment sheets.

c) Find the definition free paths for each variable by determining the DCU and DPU sets.

d) Find the minimal set of test cases that achieves all-uses. Reuse the ones from Task 3 and Project 1, if possible, and create new test cases, if necessary.

Work by analogy from Task 3 and 4 to create an appropriately sub-structured new Section 6 in your document called "Data Flow Based Tests."

Task 6

Make a brief slide presentation (with or without slides) and be ready to present your project choice in class. Don't panic.