CS 5154 Spring 2021
Homework 1

1 Deadline

This homework is due on 3/17/2021 at 9:30am EST.

2 Goals and Overview

This homework aims to reinforce the concepts that you learned in the lectures on input space partitioning.

3 Instructions

This homework is to be completed in assigned groups. You may not discuss any problem with other groups. Your submitted code and text must be created your group. The exception is the debriefing file, which should be submitted individually under the `hw-1-debriefing` assignment on CMS.

The file, `hw-1.zip`, contains files that are starting points for your solution. Download `hw-1.zip` from CMS and extract it. By the due date, you must upload a Zip file containing same-named files that contain your solution. That is, you should write your solutions in the files that we provided. Do not submit any additional files. If you add some unnecessary files (e.g., target), then you will get negative points.

We must be able to compile and run your code using **only** the files that you will submit. We will compile and run your submission on JDK 8 using Maven version `3.6.0`. You will get **no points** if your code does not compile, or if you return our code as your solution.

4 Not a Problem

If you notice bugs in any materials related to the course (e.g., slides, code, tests) at any time during the course, please report them to Owolabi, and you can get extra credit!

5 Problem 1: Input Space Partitioning for INTERSECTION (70%)

```
public static Set intersection(Set s1, Set s2)
// Effects: Return a (non null) Set equal to the intersection of sets s1 and s2
//             A null argument is treated as an empty set
Characteristic 1: Type of s1
    Block 1: s1 = null
    Block 2: s1 = {}
    Block 3: s1 has at least one element
Characteristic 3: Relation between s1 and s2
    Block 1: s1 and s2 represent the same set
    Block 2: s1 is a subset of s2
    Block 3: s2 is a subset of s1
    Block 4: s1 and s2 do not have any elements in common
```

Answer the following questions about the `intersection()` method. Put your solution in the file, `hw-1/prob1.txt`:

(a) Does the partition "Type of s1" satisfy the completeness property? If not, give a value for s1 that does not fit in any block. Does the partition "Type of s1" satisfy the disjointness property? If not, give a value for s1 that fits in more than one block.

(b) Does the partition "Relation between s1 and s2" satisfy the completeness property? If not, give a pair of values for s1 and s2 that does not fit in any block. Does the partition "Relation between s1 and s2" satisfy the disjointness property? If not, give a pair of values that fits in more than one block.

(c) Describe the input domain for the `intersection()` method.

(d) Change the blocks for the partitions "Type of s1" and "Relation between s1 and s2" such that they do not suffer from any disjointness or completeness problems, but do not just use partitions with two blocks (effectively "true" and "false").

(e) Create a partition "Type of s2" analogous to "Type of s1". Choose a representative input for each block from the three partitions "Type of s1", "Type of s2", and "Relation between s1 and s2".

(f) Describe the constraints among the three partitions used in (e) above.

Construct test cases that obey the constraints among the three partitions and satisfy the following criteria. Each test case should have two input sets and an expected output. In addition to writing your answers to (g)–(j) in `hw-1/prob1.txt`, create JUnit tests for your answers in the respective `Test*.java` files in the `hw-1/problem1` Maven project.

(g) All Combinations

(h) Each Choice, but not Pair-wise

(i) Pair-wise, but not All Combinations

(j) 3-wise

6 Problem 2: Input Space Partitioning for BOUNDEDQUEUE (25%)

Derive input space partitioning test inputs for the `BoundedQueue` class with methods that have the following signatures:

```
public BoundedQueue (int capacity); // The maximum number of elements
public void enQueue (Object X);
public Object deQueue ();
public boolean isEmpty ();
public boolean isFull ();
```

Assume the usual semantics for a queue with a fixed, maximal capacity. Try to keep your partitioning simple—choose a small number of partitions and blocks. Provide your answers in the file, `hw-1/prob2.txt`.

(a) List all of the input variables, including the state variables.

(b) Define characteristics of the input variables. Make sure you cover all input variables.

(c) Partition the characteristics into blocks. Designate one block in each partition as the "Base" block.

(d) Define values for each block.

(e) Define a set of test cases that satisfies Base Choice Coverage (BCC). Write your tests with the values from the previous step. Be sure to include the test oracles.

7 [EXTRA CREDIT] Problem 3: Write code that computes pair-wise coverage (20%)

***** NOTE: This is a challenging problem. ***** Write a method that computes a list of tuples that satisfy "Pair-wise" coverage for a given list of partitions. The number of tuples that you compute should be no more than the theoretical maximum of "Pair-wise" coverage that we discussed in class, and must not be equal to the number of tuples computed by "All Combinations" coverage for any set of input partitions. You will only get full credit if your solution produces a number of tuples that is less than the theoretical maximum of "Pair-wise" coverage. Your solution should be in the Java files in `hw-1/problem3`, which already contains a Maven project with the code skeleton. Test your method.

8 Debriefing (5%)

Answer the questions in the file, `hw-1/debriefing.txt`. Each member of your group must fill and submit this file individually in the `hw-1-debriefing` homework on CMS. Please follow the instructions at the top of the file.

9 Deliverable

Put your solutions to the above tasks in a `hw-1-solution.zip` file and upload it to CMS. Your Zip file must contain only the files mentioned in this homework, with the possible exception of `hw-1/debriefing.txt`. In your submitted Zip file, the solution files should be under a `hw-1` directory, just like in the provided zip. Specifically, in a directory containing only your `hw-1-solution.zip`, extracting and listing the file contents should look exactly like this unless you omit `debriefing.txt`:

```
$ unzip hw-1-solution.zip
$ ls
hw-1 hw-1-solution.zip
$ ls hw-1
debriefing.txt prob1.txt prob2.txt problem1 problem3
```