SUTD 2021 50.003 Problem Set 3

James Raphael Tiovalen

Cohort Exercise 1

The tester codes are included in the `MonkeyTestISTDAllLinks.java` and the `MonkeyTestISTDRandomLinks.java` file. The first file visits all the available links on ISTD's homepage once, while the second file visits a randomly selected link from ISTD's homepage and continues to do so forever.

Cohort Exercise 2

The tester code is included in the `LoginBotWithInvalidValidUser.java` file.

Cohort Exercise 3

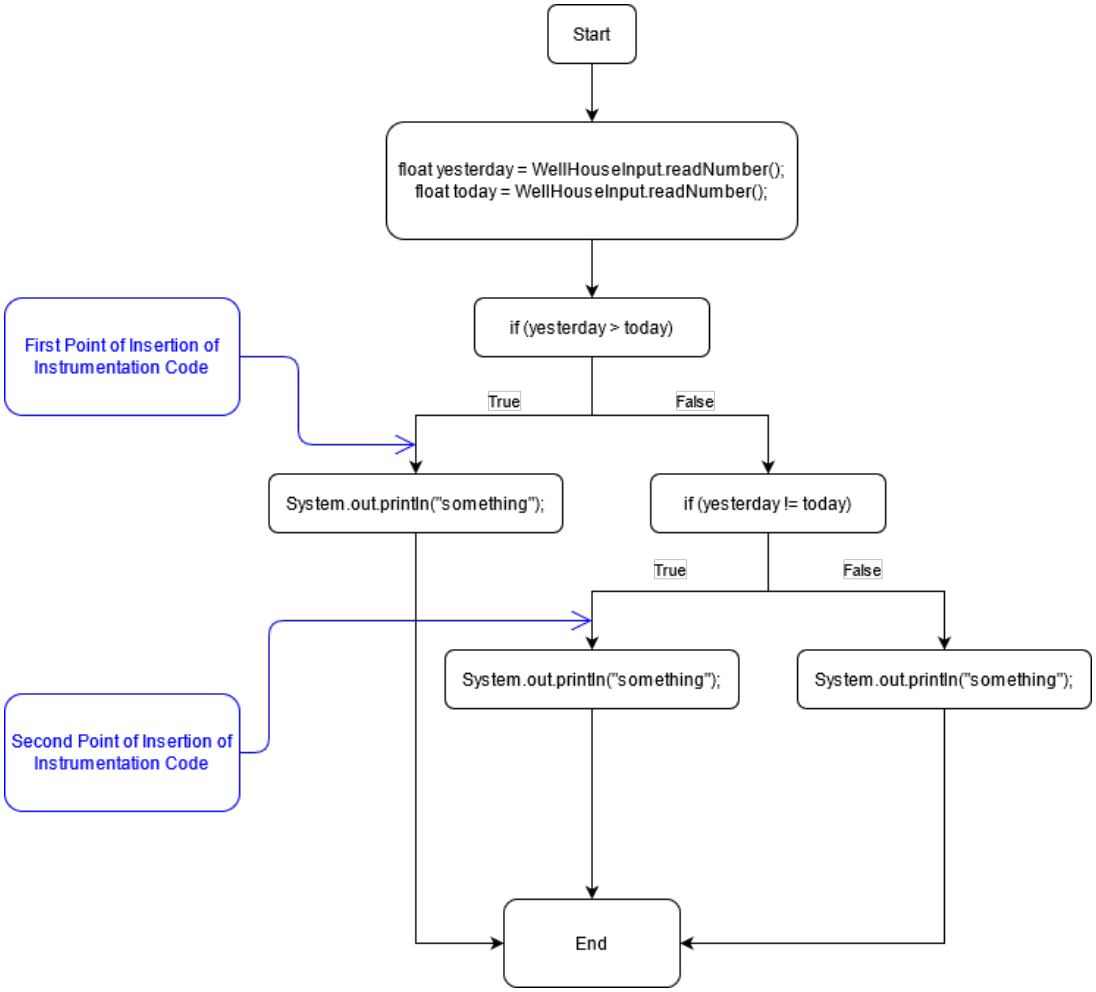
The tester code is included in the `HeaderNameFinder.java` file.

Cohort Exercise 4

The calculator grammar fuzzer code is included in the `calculator-grammar-fuzzing.py` file.

Cohort Exercise 5

The control flow graph, along with the corresponding points of insertion of instrumentation code, of the `foo()` function is presented here:



Cohort Exercise 6

The genetic algorithm's fitness function in the `FitnessCalc.java` file has been modified accordingly so as to produce palindromic strings with 64 characters instead of attempting to generate a fixed static target text. The fitness function only needs to loop until the end of the first half of the characters of the string, since the second half of the characters of the string can just be compared against the first half (i.e., the fitness function is defined to be the accumulated difference between the first half's gene and the second half's correspondingly mirrored gene). The other selection/crossover/mutation operators are unchanged.

Cohort Exercise 7

These are the resulting statistics:

- `Example.java`:
    - Conditional branch coverage: 93%.
    - Average coverage: 98%.

    - Time taken: 70266 milliseconds.
- `Example1.java`:
    - Conditional branch coverage: 100%.
    - Average coverage: 100%.
    - Time taken: 23695 milliseconds.
- `Example2.java`:
    - Conditional branch coverage: 73%.
    - Average coverage: 92%.
    - Time taken: 69628 milliseconds.

`Example1.java` takes much less time than `Example.java` and it obtains 100% coverage, which might be due to more satisfiable conditions that have a much higher probability to be satisfied by random inputs generated by EvoSuite's genetic algorithm.

`Example2.java` takes about the same time as `Example.java`. Its lower coverage percentage might be caused by the equality conditions, which are much harder to be found by EvoSuite's genetic algorithm. The equality conditions might simultaneously satisfy the conditions on several levels of the if branches, which might be considered as a lower coverage percentage by EvoSuite since the subsequent identical conditions would always be true and hence there are no possibilities of executing the other branches of that conditional statements.

Homework Question 1

The generalized fuzzer is implemented in the included `mutation-fuzzing.c` file. The input file is generalized-fuzzer-input.txt and the output file is generalized-fuzzer-output.txt.

Cohort Exercise 8

The "repeated code smell" has been removed in the `BrokenLinkFinderNoSmell.java` file.

Cohort Exercise 9

The "long method smell" has been removed and the `CreditTransaction()` method has been added in the `AccountNoSmell.java` file.

Cohort Exercise 10

The "shotgun surgery code smell" has been removed, while the `CreditTransaction()` method and the account type restriction/limitation feature has been added in the `ShootTheAccountPlus.java` file.

Cohort Exercise 11

The code in `XSSFixed.java` has been modified accordingly to catch all possible variations of XSS-styled attacks with the pattern `<script>` (including its uppercase, lowercase and Unicode variants).

Cohort Exercise 12

Since the `after()` method internally calls the `compareTo()` method, in the original code, since the `compareTo()` function is overridden, calling `super.after()` will still end up in calling the `compareTo()` method in the `Calendar` subclass. The code in `exercise4.java` has been modified accordingly in order to fix this behavior. The `CalendarTest.java` provides a test suite to check and verify whether the code in `exercise4.java` is correct or not.

Homework Question 2

The loop-free C program with 8 different paths requires the usage of conditional branches. The corresponding code filename is `homework.c`. The .smt2 test files along with their corresponding computed satisfiability models using Z3 are also included together as .txt files under the same directory with the same filename.