Problem Set 4

(10 points) Cohort Exercise 1:

Given Sorter.java and FindMaxUsingSorting.java, write a test case for method FindMax without implementing Sorter.java first.

(10 points) Cohort Exercise 2:

Often input strings are expected to be alphanumeric. Modify the simple fuzzer to generate random alphanumeric strings of length between 0 and 1024. You can use any programming language.

(10 points) Cohort Exercise 3:

Given an input string, implement a mutation operator that choose a random position in the string and swap the adjacent characters. This means if SUTD is an input string, and 2 is chosen as the random position, and then the output should be SUDT. Careful about the string length bound check.

(10 points) Cohort Exercise 4:

Implement a generalized fuzzer that will take a file, read each line of the file, randomly choose a mutation operator (swap, bit flip or trim) and produce a different file with the modified lines. This means for each input file, the fuzzer will produce one output file, where each line is modified with a random mutation operator. Choose any programming language. You must your program modular so that more mutation operators can be added easily.

(5 points) Cohort Exercise 5:

Consider the calculator grammar shown in class. Draw the derivation tree for input (23 * 56) / (1.2 + 2).

(20 points) Cohort Exercise 6:

Use any programming language to implement a fuzzer that will randomly generate inputs to the calculator conforming to the grammar. Make your program modular in the sense that it can work with any grammar, as long as it is available.

Hint: Start with the initial rule S := Expr and at each point, apply a rule at random. For example, randomly choose any of the rule Expr := Term, Expr := Expr + Term or Expr – Term in the next step. Continue until a valid expression for the calculator is obtained.

(10 points) Cohort Exercise 7:

Test the new istd.sutd.edu.sg webpage by randomly clicking links. The process may go on forever.

(5 points) Cohort Exercise 8:

Extend the food ordering bot to show a list of restaurants in your area once you are logged in.

(10 points) Cohort Exercise 9:

Write a test to do the following: Send 20 invalid usernames to Google login form (be innovative about the invalid user names) and then login with your user name and password.

(10 points) Cohort Exercise 10:

Write a test to check whether all web pages *directly* reachable from a given webpage have titles. The test will fail if any such directly reachable webpage has an empty title. Hint: Use getTitle() from the web driver.