CIS 285: Software Engineering Tools

University of Michigan – Dearborn

Lab 10

Unit Testing and Git

Location: At home

Due Date: Thursday 04/13 – Submission is due by 11:59 pm on Canvas

This laboratory exercise engages you in unit testing-based development of a simple Java method. You don’t need to be a Java expert to complete this exercise.

**Laboratory tasks**:

1. Unit testing with JUnit for the method “basicSelectionSort” in SelectionSort.java class
   1. Create a new Java Project in Eclipse
   2. Add a new Java class “SelectionSort.java” in the project (initial source code in the next page)
   3. Add a JUnit Test Case “testSelectionSort.java” in the project
   4. Develop and execute the following unit tests
   5. testPositives – testing a list of all positive integers
   6. testNegatives – testing a list of all negative integers
   7. testMixed – testing a list containing positive, negative and zeros.
   8. testDuplicates – testing a list containing one or more duplicate number, for both positive and negative numbers
2. Git for Configuration management
   1. Create a new local repository containing original Selectionsort.java and testSelectionsort.java
   2. Create 2 branches, one working on testPositives and testNegatives methods, the other one working on testMixed and testDuplicates in testSelectionsort.java
   3. Use atom to edit codes in both branches
   4. After both branches pass the test, merge them into master and run test on master branch to make sure all test pass.
   5. Go to github.com to create a new repository
   6. Push the master branch to the github repository

**Report submission**:

Create a MS Word or PDF document containing the following:

1. Unit testing with JUnit
   1. Source code of initial SelectionSort.java class
   2. Source code of unit tests created and executed
   3. Outputs of unit tests. Include both failed and passed unit tests. In the case of failed unit tests, indicate what was wrong and what was done to fix the problem.
   4. Final outputs of all unit tests showing successful pass for all
   5. Source code of final SelectionSort.java class
2. Git for Configuration management
   1. Show screen shots github that contains Selectionsort.java and testSelectionsort.java
   2. Local git folder

You are to submit your MS Word or PDF report and other required items on Canvas as a ZIP file.

Source Codes:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**public** **class** SelectionSort {

**private** **int** temp;

/\*\* Creates a new instance of SelectionSort \*/

**public** SelectionSort() {

}

/\* A simple SelectionSort algorithm

\* pre-condition:

\* post-condition:

\* inputs:

\* outputs:

\* special conditions:

\*/

**public** **int**[] basicSelectionSort(**int**[] x) {

**for** (**int** i = 0; i < x.length; ++i) {

**for** (**int** j= i+1; j < x.length; ++j) {

**if** (x[i] > x[j]) {

temp = x[i];

x[i] = x[j];

temp = x[j];

}

} // end of inner for loop

} // end of outer for loop

**return** x;

} // end of basicSelectionSort method

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*

\* testSelectionSort.java

\*

\* To change this template, choose Tools | Template Manager

\* and open the template in the editor.

\*/

import static org.junit.Assert.\*;

import org.junit.Assert;

import org.junit.Test;

public class testSelectionSort {

@Test

public void test() {

testPositive();

testNegative();

testMixed();

testDuplicates();

}

public testSelectionSort() {

}

public void testPositive(){

int[] arr = new int[5];

arr[0] = 8;

arr[1] = 9;

arr[2] = 7;

arr[3] = 10;

arr[4] = 2;

int[] Sortedarr = new int[5];

Sortedarr[0] = 2;

Sortedarr[1] = 7;

Sortedarr[2] = 8;

Sortedarr[3] = 9;

Sortedarr[4] = 10;

/\*\* add tests to check for this unit test \*\*/

}

public void testNegative(){

/\*\* Test data contains negative values only \*\*/

}

public void testMixed(){

/\*\* Test data contains with both positive, negative and zeros \*\*/

}

public void testDuplicates(){

/\*\* Test data contains duplicates \*\*/

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*