Ryerson University

Faculty of Engineering & Architectural Science

Department of Biomedical, Computer, and Electrical Engineering

Course Number	COE891
Course Title	Software Testing and Quality Assurance
Semester/Year	Winter2024
Instructor	Dr. Reza Samavi
Section No.	01
Group No.	N/A
Submission Date	Feb 11, 2024
Due Date	Feb 12, 2024

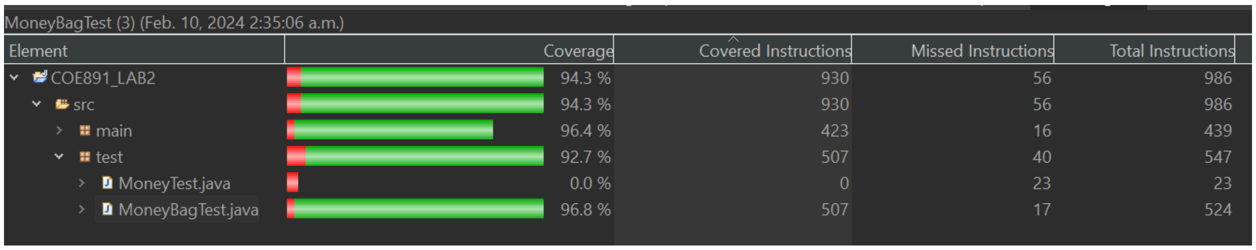
Lab/Tut Assignment NO.	2

Assignment Title	Coverage-based Test Design and Input Domain Model

Name	**Student ID**	**Signature***
Astha Patel	501040209	[illegible]

Q1:

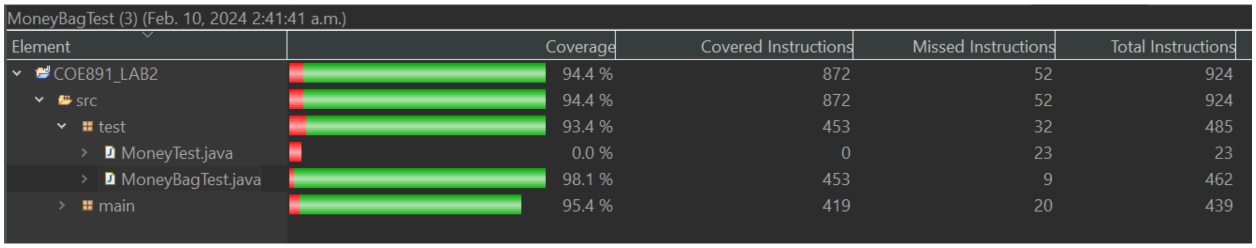
The two following figures, Figure 1 and Figure 2, display the MoneyBagTest.java test class before and after modification to achieve 100% test coverage. The coverage of the test class before any modification results in a 96.8% meanwhile the coverage after commenting out the testMoneyBagEquals() test method.



MoneyBagTest (3) (Feb. 10, 2024 2:35:06 a.m.)

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
COE891_LAB2	94.3 %	930	56	986
src	94.3 %	930	56	986
main	96.4 %	423	16	439
test	92.7 %	507	40	547
MoneyTest.java	0.0 %	0	23	23
MoneyBagTest.java	96.8 %	507	17	524

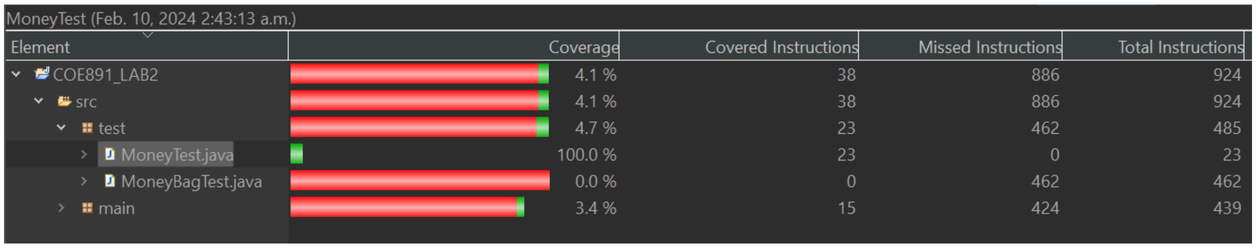
Figure 1: Before MoneyBagTest.java modification



MoneyBagTest (3) (Feb. 10, 2024 2:41:41 a.m.)

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
COE891_LAB2	94.4 %	872	52	924
src	94.4 %	872	52	924
test	93.4 %	453	32	485
MoneyTest.java	0.0 %	0	23	23
MoneyBagTest.java	98.1 %	453	9	462
main	95.4 %	419	20	439

Figure 2: After MoneyBagTest.java modification



MoneyTest (Feb. 10, 2024 2:43:13 a.m.)

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
COE891_LAB2	4.1 %	38	886	924
src	4.1 %	38	886	924
test	4.7 %	23	462	485
MoneyTest.java	100.0 %	23	0	23
MoneyBagTest.java	0.0 %	0	462	462
main	3.4 %	15	424	439

Figure 3: MoneyTest coverage

Q2:

```
public int func(int a, int b) {
       if (b > a) {
              b = b - a;
              return b; }
       else if (a > b){
              b = a - b;
              return b; }
       else
              return 0;}
 }
```

1.) a = 2 , b = 3

- 10 total statements
- Line coverage
  - Lines covered : 1,2,3,4,10
  - (# of statements covered / total number of statements) % 100 = (5/10)*100 = 50% coverage

- Branch coverage
  - 4 total branches
  - 1st branch covered
    - If (b > a)
  - (# of branches covered / total number of branches) % 100 = (1/3)*100 = 33.33%

2.) a = 3 , b = 2

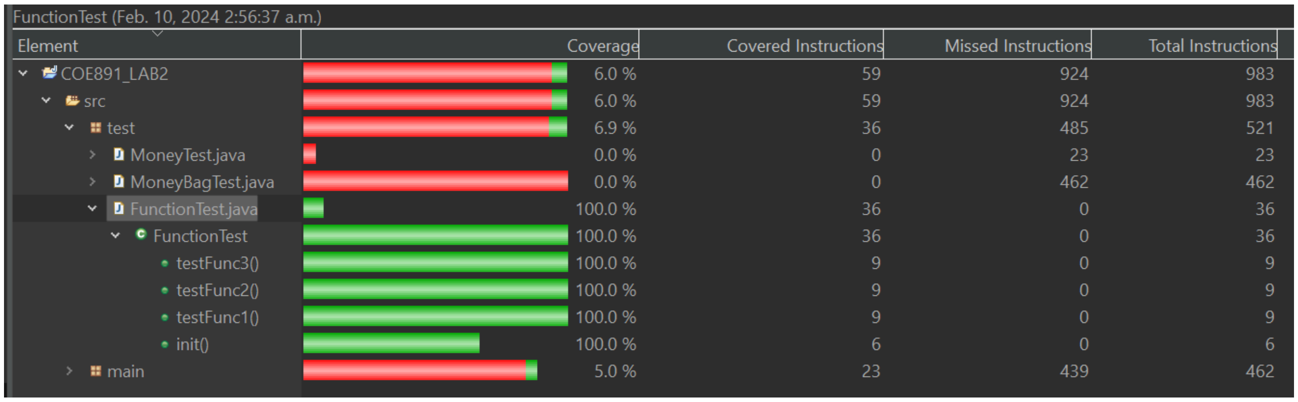
- 10 total statements
- Line coverage
  - Lines covered : 1,2,3,4,10
  - (# of statements covered / total number of statements) % 100 = (5/10)*100 = 50% coverage

- Branch coverage
  - 4 total branches
  - 2ndt branch covered
    - If (a > b)
  - (# of branches covered / total number of branches) % 100 = (1/3)*100 = 33.33%

3.) a = 2 , b = 2

- 10 total statements
- Line Coverage
  - Lines covered : 1,8,9,10
  - (# of statements covered / total number of statements) % 100 = (4/10)*100 = 40% coverage
- Branch coverage
  - 4 total branches
  - 3rd branch covered
    - else …

- (# of branches covered / total number of branches) % 100 = (1/3)*100 = 33.33%



FunctionTest (Feb. 10, 2024 2:56:37 a.m.)

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
COE891_LAB2	6.0 %	59	924	983
src	6.0 %	59	924	983
test	6.9 %	36	485	521
MoneyTest.java	0.0 %	0	23	23
MoneyBagTest.java	0.0 %	0	462	462
FunctionTest.java	100.0 %	36	0	36
FunctionTest	100.0 %	36	0	36
testFunc3()	100.0 %	9	0	9
testFunc2()	100.0 %	9	0	9
testFunc1()	100.0 %	9	0	9
init()	100.0 %	6	0	6
main	5.0 %	23	439	462

Figure 4: Coverage for FunctionTest.java test class

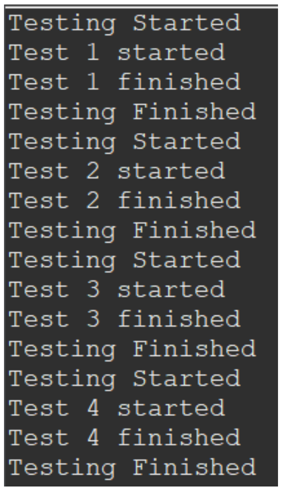
Coverage percentage for FunctionTest.java test class is able to achieve 100% coverage for the test class by adding the test case with conditions, a = 2 and b = 2. This covers the third branch in the java class that handles equal a and b values.

Q3:

The following code statement displays the java method for TriClass.java class.

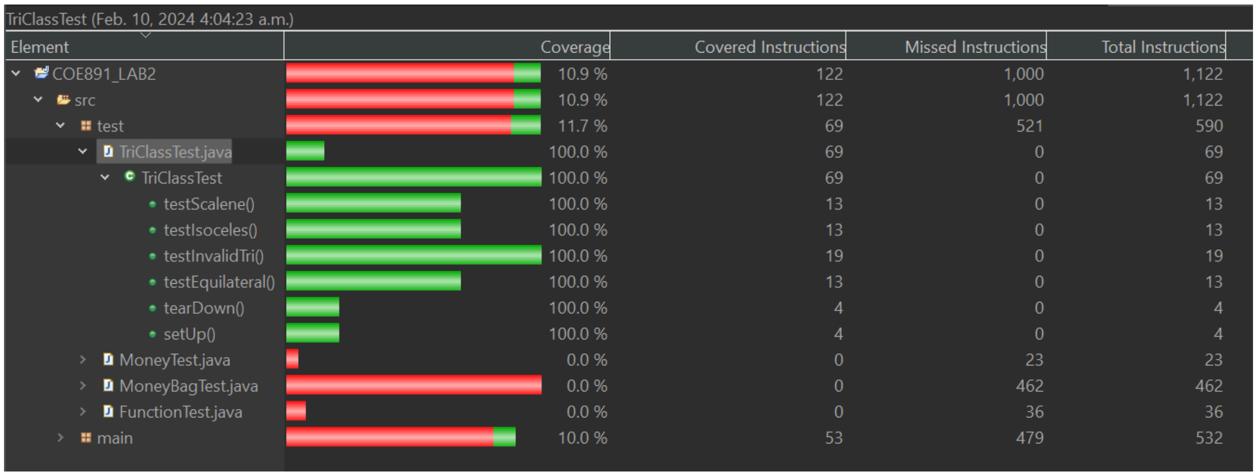
```java
public static String classify(int x, int y, int z)
       {
              if ( !(x + y > z) || !(y + z > x) || !(z + x > y)) {
                     return "Invalid";
              }
              if (x == y && y == z) {
                     return "Equilateral";
              } else if ((x == y && y != z && x != z) || (y == z && z != x && x != y) || (z
== x && x != y && y != z)){
                     return "Isosceles";
              } else if (x != y && y != z && x != z) {
                     return "Scalene";
              } else {
                     return "Invalid";
              }

       }
```



```
Testing Started
Test 1 started
Test 1 finished
Testing Finished
Testing Started
Test 2 started
Test 2 finished
Testing Finished
Testing Started
Test 3 started
Test 3 finished
Testing Finished
Testing Started
Test 4 started
Test 4 finished
Testing Finished
```

Figure 5: System print messages

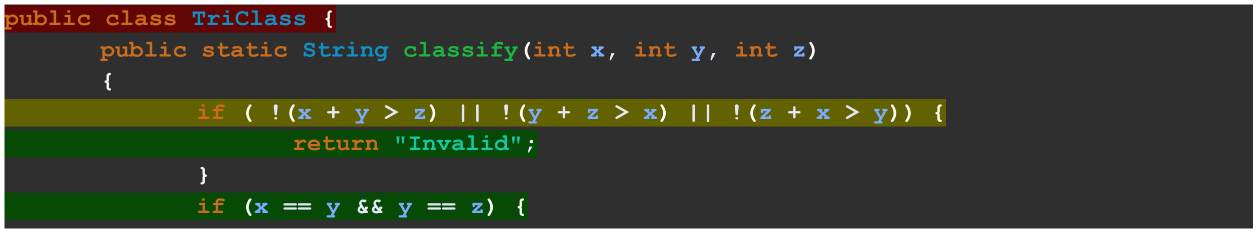


TriClassTest (Feb. 10, 2024 4:04:23 a.m.)

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
COE891_LAB2	10.9 %	122	1,000	1,122
src	10.9 %	122	1,000	1,122
test	11.7 %	69	521	590
TriClassTest.java	100.0 %	69	0	69
TriClassTest	100.0 %	69	0	69
testScalene()	100.0 %	13	0	13
testIsoceles()	100.0 %	13	0	13
testInvalidTri()	100.0 %	19	0	19
testEquilateral()	100.0 %	13	0	13
tearDown()	100.0 %	4	0	4
setUp()	100.0 %	4	0	4
MoneyTest.java	0.0 %	0	23	23
MoneyBagTest.java	0.0 %	0	462	462
FunctionTest.java	0.0 %	0	36	36
main	10.0 %	53	479	532

Figure 6: Coverage for TriClassTest.java test class.

The following code snippet showcases the coverage areas for the TriClass.java when tested in the TriClassTest.java test class. Red indicates no coverage which is usually for unused statements and any methods or classes that are not run as a part of the test. Yellow indicates failed statements which is usually due to any conditional branches since only one branch will succeed during a specific test case for each branch statement testing. Green indicates passed test, in the following code snippet, the return statements are green because every test case that reaches those return statements will be due to passing the branch statement. Hence, a passed test case.



```java
public class TriClass {
      public static String classify(int x, int y, int z)
      {
            if ( !(x + y > z) || !(y + z > x) || !(z + x > y)) {
                  return "Invalid";
            }
            if (x == y && y == z) {
```

```
            return "Equilateral";
        } else if ((x == y && y != z && x != z) || (y == z && z != x && x != y)
|| (z == x && x != y && y != z)){
            return "Isosceles";
        } else if (x != y && y != z && x != z) {
            return "Scalene";
        } else {
            return "Invalid";
        }
    }
}
```