Assignment 13

Rasika Mohod
rmohod@gmu.edu
G01044774

I tried working with Major mutation testing tool but could not make it work successfully. I also tried Pex and Moles and Code Digger tools for generating unit tests with Visual studio, but both were compatible with very old versions of Visual Studio (VS 10 and 12). I tried and installed different VS versions but could not find any VS compatible for these tools.

So as a solution to this, I have worked on Junit-Tools for this assignment. It is a tool to generate JUnit test-elements (packages, classes, methods, test-cases, mocks) depending on an existing java-class and logic.

The program on which tool was run is simple class with two methods to compute if the given number if even or odd. The code for the program is given as follows:

Computation.java:

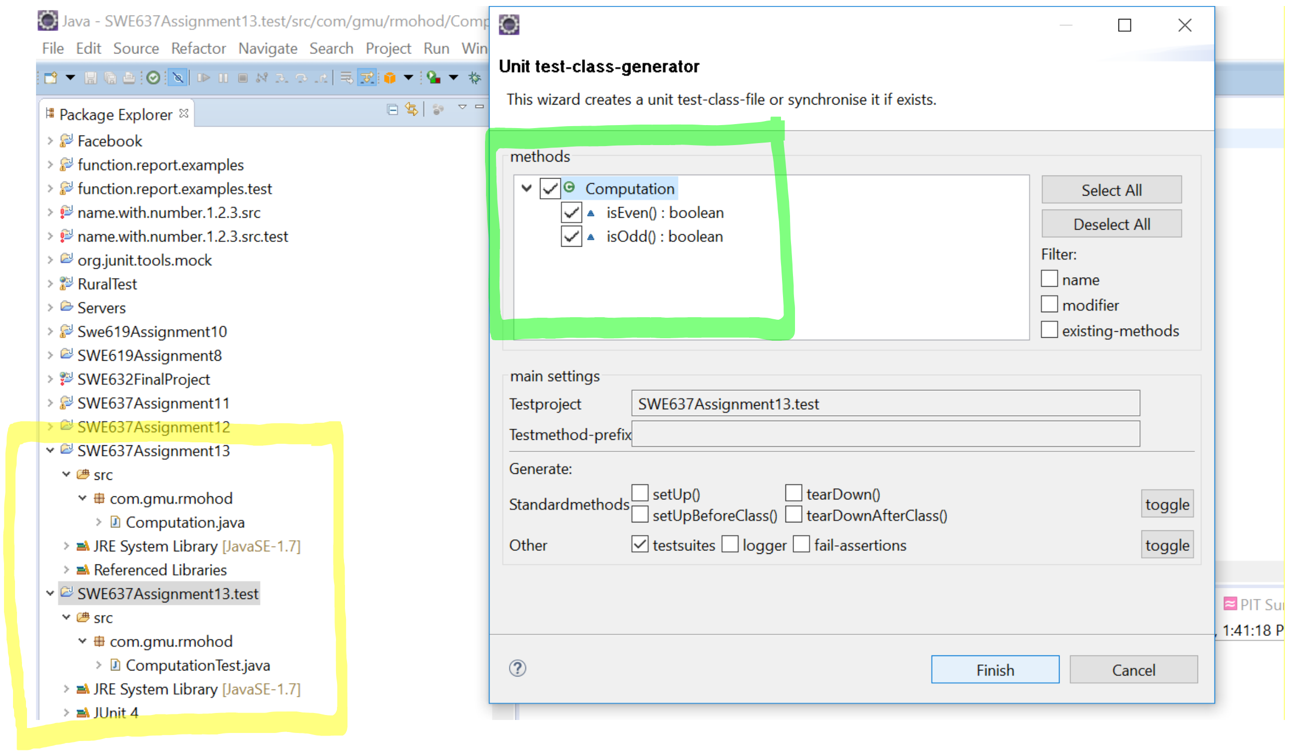
```
package com.gmu.rmohod;
/**
 * Class to compute if a number is even or odd
 *
 * @author Rasika
 */
public class Computation {
	int number;
	Computation(int x)
	{
		number = x;
	}
	boolean isEven()
	{
		if(number%2==0)
			return true;
		else
			return false;
	}
	boolean isOdd()
	{
		if(number%2==1)
			return true;
		else
			return false;
	}
}
```

JUnit-Tools provides methods to generate unit test for desired methods and also mocks for the same.
Given below is the image of creation of unit tests for two methods of Computation class.

Yellow marked region shows two projects:

1. SWE637Assignment13 which contains existing logic
2. SWE637Assignment13.test which will contain the tests generated by JUnit-Tools

Green marked region shows the methods selected for which tests will be generated by JUnit-Tools plugin.


Java - SWE637Assignment13.test/src/com/gmu/rmohod/Comp
File Edit Source Refactor Navigate Search Project Run Win
Package Explorer

Facebook
function.report.examples
function.report.examples.test
name.with.number.1.2.3.src
name.with.number.1.2.3.src.test
org.junit.tools.mock
RuralTest
Servers
Swe619Assignment10
SWE619Assignment8
SWE632FinalProject
SWE637Assignment11
SWE637Assignment12
SWE637Assignment13
  src
    com.gmu.rmohod
      Computation.java
  JRE System Library [JavaSE-1.7]
  Referenced Libraries
SWE637Assignment13.test
  src
    com.gmu.rmohod
      ComputationTest.java
  JRE System Library [JavaSE-1.7]
  JUnit 4

Unit test-class-generator

This wizard creates a unit test-class-file or synchronise it if exists.

methods

- [✓] Computation
  - [✓] isEven() : boolean
  - [✓] isOdd() : boolean

Filter:
[ ] name
[ ] modifier
[ ] existing-methods

main settings

Testproject: SWE637Assignment13.test
Testmethod-prefix:

Generate:

Standardmethods: [ ] setUp() [ ] tearDown() [ ] setUpBeforeClass() [ ] tearDownAfterClass()

Other: [✓] testsuites [ ] logger [ ] fail-assertions

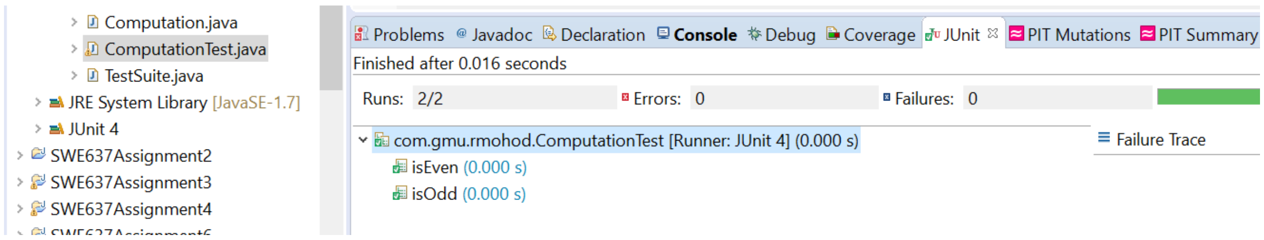
Given below is the test class with unit tests generated by JUnit-Tools:

<u>CompuatationTest.java</u>

```
package com.gmu.rmohod;

import mockit.Deencapsulation;
import org.junit.Test;
import org.junit.tools.configuration.base.MethodRef;

public class ComputationTest {

	private Computation createTestSubject() {
		return new Computation(0);
	}
```

```
    @MethodRef(name = "isEven", signature = "()Z")
    @Test
    public void isEven() throws Exception {
        Computation testSubject;
        boolean result;

        // default test
        testSubject = createTestSubject();
        result = Deencapsulation.invoke(testSubject, "isEven");
    }

    @MethodRef(name = "isOdd", signature = "()Z")
    @Test
    public void isOdd() throws Exception {
        Computation testSubject;
        boolean result;

        // default test
        testSubject = createTestSubject();
        result = Deencapsulation.invoke(testSubject, "isOdd");
    }

}
```

JUnit Output:

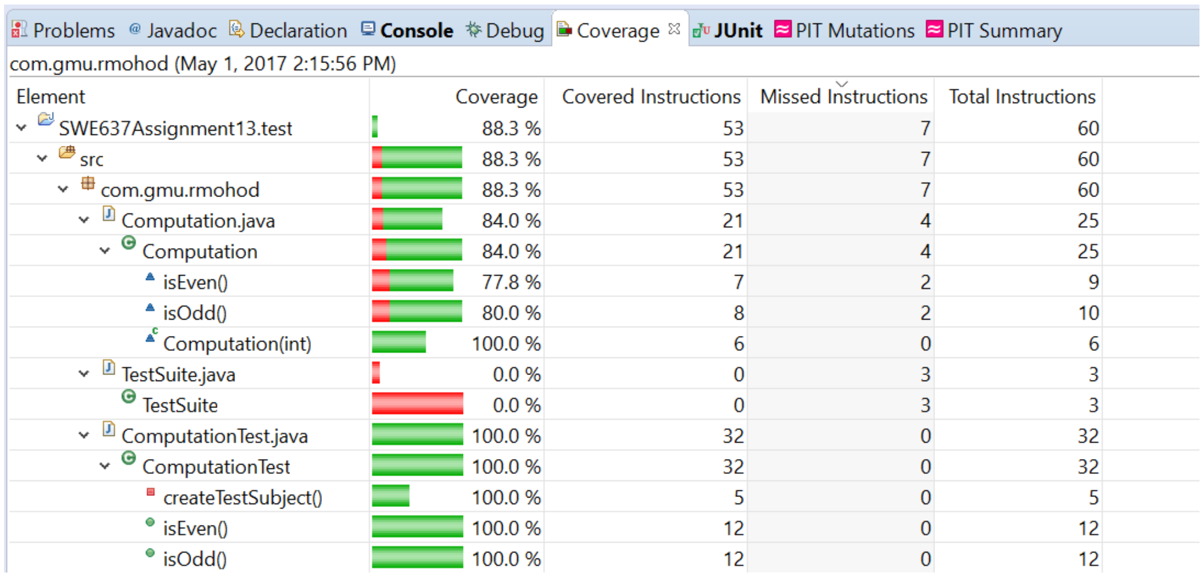


```
Computation.java
ComputationTest.java
TestSuite.java
JRE System Library [JavaSE-1.7]
JUnit 4
SWE637Assignment2
SWE637Assignment3
SWE637Assignment4
```

Problems | Javadoc | Declaration | Console | Debug | Coverage | JUnit | PIT Mutations | PIT Summary

```
Finished after 0.016 seconds
Runs: 2/2        Errors: 0        Failures: 0

com.gmu.rmohod.ComputationTest [Runner: JUnit 4] (0.000 s)
    isEven (0.000 s)
    isOdd (0.000 s)
```

Test Coverage:



Problems | Javadoc | Declaration | Console | Debug | Coverage | JUnit | PIT Mutations | PIT Summary

com.gmu.rmohod (May 1, 2017 2:15:56 PM)

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
SWE637Assignment13.test	88.3 %	53	7	60
src	88.3 %	53	7	60
com.gmu.rmohod	88.3 %	53	7	60
Computation.java	84.0 %	21	4	25
Computation	84.0 %	21	4	25
isEven()	77.8 %	7	2	9
isOdd()	80.0 %	8	2	10
Computation(int)	100.0 %	6	0	6
TestSuite.java	0.0 %	0	3	3
TestSuite	0.0 %	0	3	3
ComputationTest.java	100.0 %	32	0	32
ComputationTest	100.0 %	32	0	32
createTestSubject()	100.0 %	5	0	5
isEven()	100.0 %	12	0	12
isOdd()	100.0 %	12	0	12

Given below is the mock generated by JUnit-Tools:

ComputationMock.java

```
package com.gmu.rmohod;

import javax.annotation.Generated;

import mockit.Invocation;
import mockit.Mock;
import mockit.MockUp;

import org.junit.tools.configuration.base.MethodRef;

/** Mock for { @link Computation } */
@Generated(value = "org.junit-tools-1.0.5")
public class ComputationMock extends MockUp<Computation> {

	private boolean isEvenMocked = false;
	private int isEvenExecutions = 0;
	private boolean isEvenReturnValue = false;
	private boolean isOddMocked = false;
	private int isOddExecutions = 0;
	private boolean isOddReturnValue = false;

	public static ComputationMock create() {
		return new ComputationMock();
	}

	@MethodRef(name = "isEven", signature = "()Z")
	@Mock
	boolean isEven(Invocation inv) {
		isEvenExecutions++;
		if (isEvenMocked) {
			return isEvenReturnValue;
		}
		return inv.proceed();
	}

	public void setUpMockIsEven(boolean returnValue) {
		isEvenReturnValue = returnValue;
		isEvenMocked = true;
		isEvenExecutions = 0;
	}

	public int getIsEvenExecutions() {
		return isEvenExecutions;
	}

	public boolean isIsEvenExecuted() {
		return isEvenExecutions > 0;
	}

	public void resetMockIsEven() {
		isEvenMocked = false;
		isEvenExecutions = 0;
	}

	@MethodRef(name = "isOdd", signature = "()Z")
	@Mock
	boolean isOdd(Invocation inv) {
		isOddExecutions++;
		if (isOddMocked) {
			return isOddReturnValue;
		}
		return inv.proceed();
	}

	public void setUpMockIsOdd(boolean returnValue) {
		isOddReturnValue = returnValue;
		isOddMocked = true;
		isOddExecutions = 0;
	}

	public int getIsOddExecutions() {
		return isOddExecutions;
	}

	public boolean isIsOddExecuted() {
		return isOddExecutions > 0;
	}

	public void resetMockIsOdd() {
		isOddMocked = false;
		isOddExecutions = 0;
	}

	public void resetAllMocks() {
		resetMockIsEven();
		resetMockIsOdd();
	}
}
```