

SE 4367: Software Testing Verification Validation and Quality Assurance

Assignment-2

Out: Sept. 17th; Due: 11:59pm, Sept. 26th

Submission: *.java via eLearning

Maximum points: 20

You can find the original sample project (SE4367-ASM) in the “tutorial code” folder on eLearning. Note that the original sample project contains various coding examples for using ASM. The submitted file should contain only one Java file, e.g., the updated “MethodTransformVisitor.java”. Note that your code has to be executable without any compilation error when the TA replace the original “MethodTransformVisitor.java” with your file, otherwise it will not be graded.

In project *SE4367-ASM*, “MethodTransformVisitor.java” modifies the method bytecode to collect method level coverage information. Extend that file to also collect the statement level coverage information. Note that as I showed in the class, when you execute the following command, “TransformClassFile” will invoke the class level instrumentation in “ClassTransformVisitor”, which in turn invokes the method-level instrumentation at “MethodTransformVisitor”.

```
java -cp ../lib/* TransformClassFile Example.class.bak Example.class
```

After your extension, the above command should be able to collect both method and statement coverage precisely through byte-code manipulation. For example, for the following code “Example.java” file (included in the project), after instrumentation your tool should be able to trace all the executed lines correctly.

Hint: You shall override the “visitLineNumber” and “visitLabel” methods in “MethodTransformVisitor.java” to achieve the desired goal.

```
public class Example {
    public static void main(String[] args) {
        m1();
        m2();
    }
    public static void m1() {
        int a=1;
        if (a < 0)
            a++; else a--;
    }
    public static void m2() {
        m1();
        m1();
    }
}
```

The expected output when executing the instrumented code should be:

```
main executed
line 0 executed
line 3 executed
m1 executed
line 0 executed
line 7 executed
line 7 executed
line 8 executed
line 9 executed
line 10 executed
line 3 executed
line 4 executed
m2 executed
line 0 executed
line 12 executed
m1 executed
line 0 executed
line 7 executed
line 7 executed
line 8 executed
line 9 executed
line 10 executed
line 12 executed
line 13 executed
m1 executed
line 0 executed
line 7 executed
line 7 executed
line 8 executed
line 9 executed
line 10 executed
line 13 executed
line 14 executed
line 4 executed
line 5 executed
```