软件测试上机报告

第四次上机作业

学　　院___智能与计算学部__
专　　业___软件工程_____
姓　　名___王兆盟_____
学　　号___3017218072_____
年　　级___2017 级_____
班　　级___软件工程一班____

1、Experiment Requirement

Tasks:

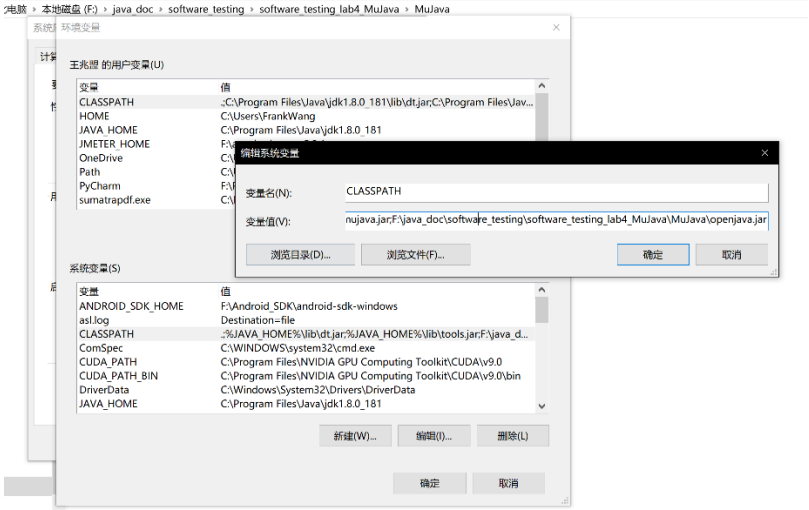
1. Install MuJava. The instruction of how to install and use Mujava can be seen in https://cs.gmu.edu/~offutt/mujava/ .
2. Two small programs are given for your task. BubbleSort.java is an implementation of bubble sort algorithm and BackPack.java is a solution of 01 backpack problem. Try to generate Mutants of 2 given programs with MuJava.
3. Write testing sets for 2 programs with Junit, and run mutants on the test sets with MuJava.

Requirements for the experiment:

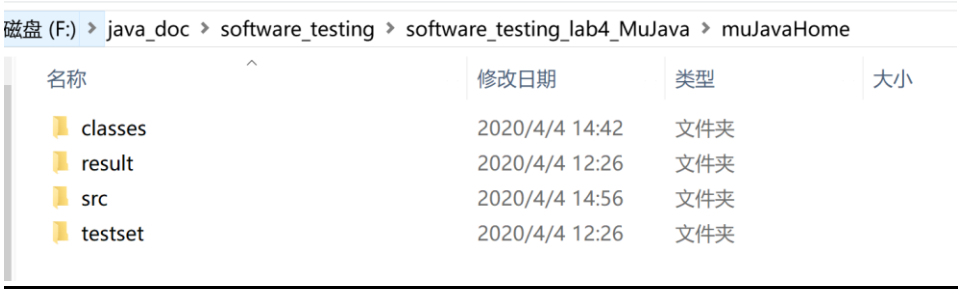
1. Finish the tasks above individually.
2. Check in your java code to github or gitee.
3. Post your experiment report to “智慧树”, the following information should be included in your report:
   a) The brief description that you install MuJava
   b) Steps for generating Mutants
   c) Steps for making test sets and running mutants.
   d) Your mutants result (The number of live mutants, killed mutants, etc.)

2、Source Code and Experiment Process

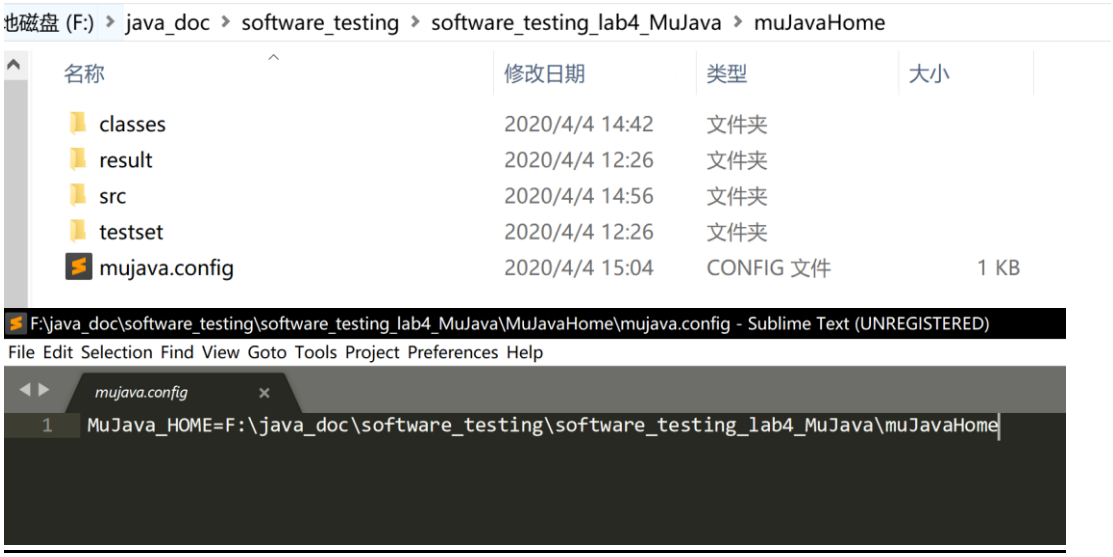
Firstly, I installed the MuJava unit by adding its path to the environment variables “CLASSPATH”.



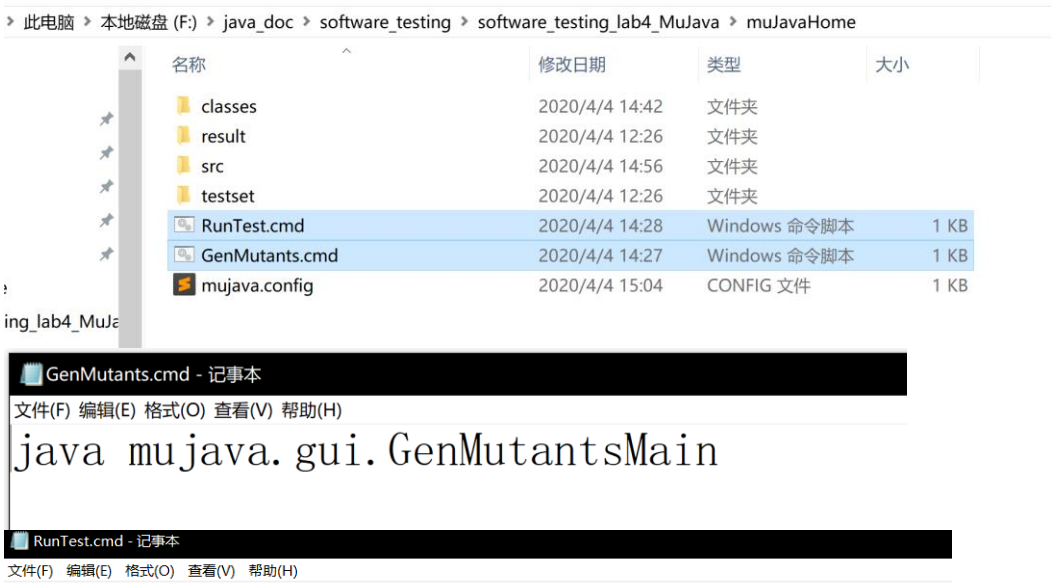
Then under the guidance of the website (https://cs.gmu.edu/~offutt/mujava/), I build the project structure as shown below, via creating those four folders in a new folder “muJavaHome”.



Next, write the config file of MuJava.



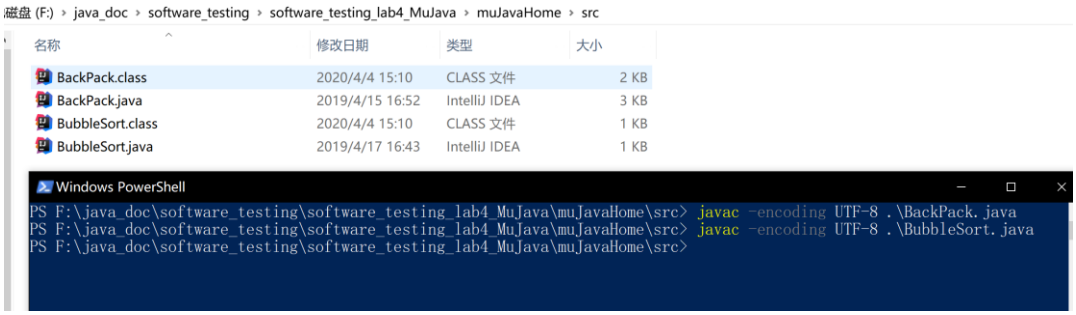
Write those two “.cmd” file to execute the command.



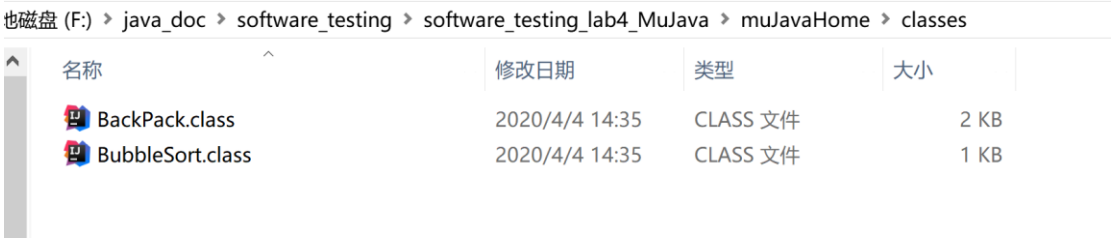

```
java mujava.gui.RunTestMain > TestResult.txt
```

This command shown above configured the test result save as TestResult.tx in the project path.

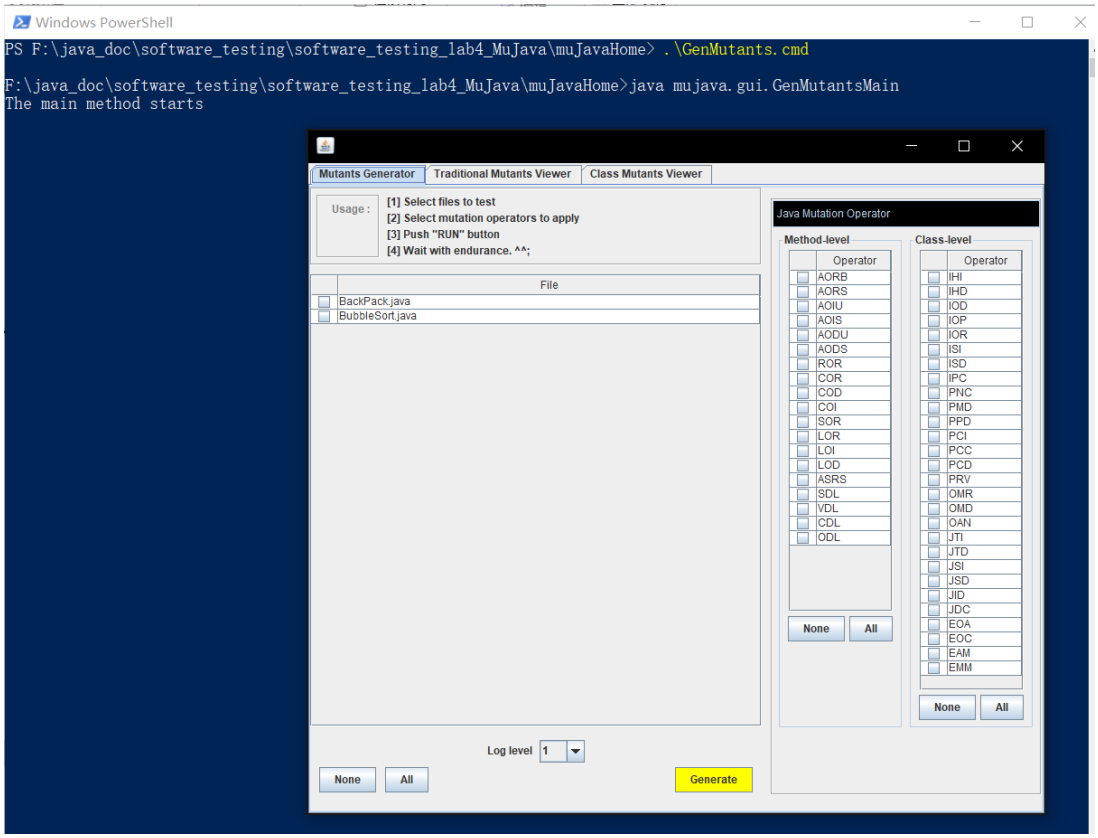
And then move those two script to be test(BubbleSort.java, BackPack.java) into the “src” folder, using the command “javac” to compile the .java files to generate .class files. In this step, due to the Chinese annotation in those file ware encoded in “UTF-8”, so we are supposed to add the parameter “-encoding UTF-8”.



Next, move those two class files into to the “classes” folder.

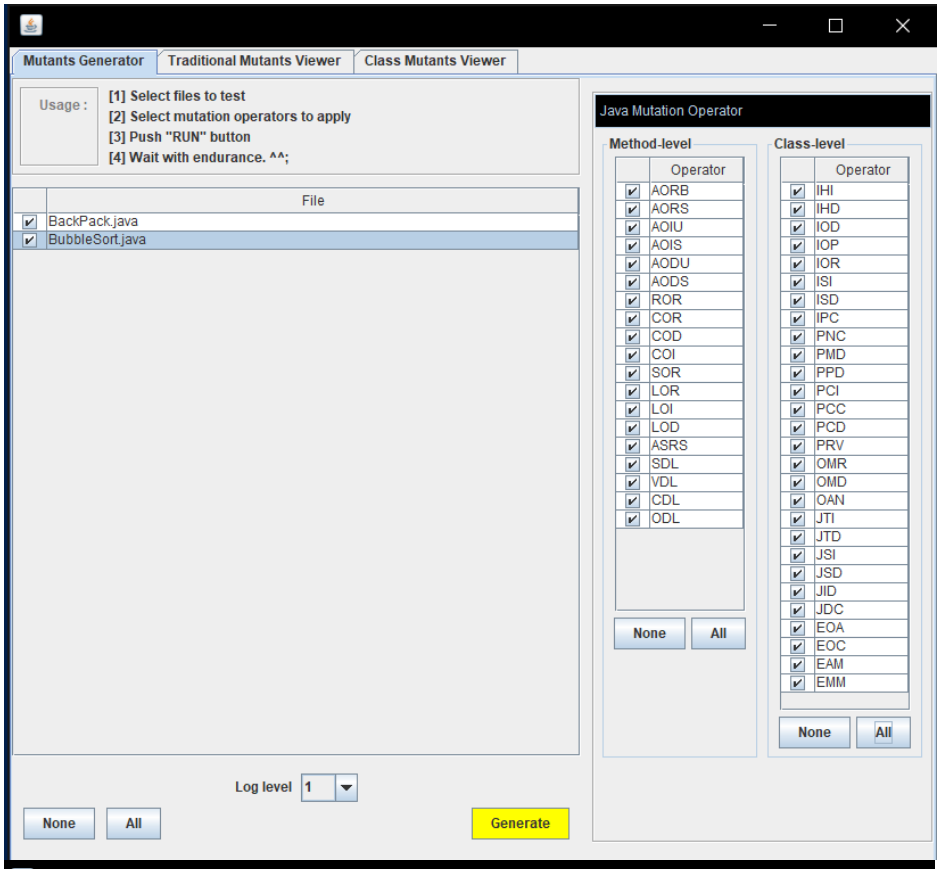


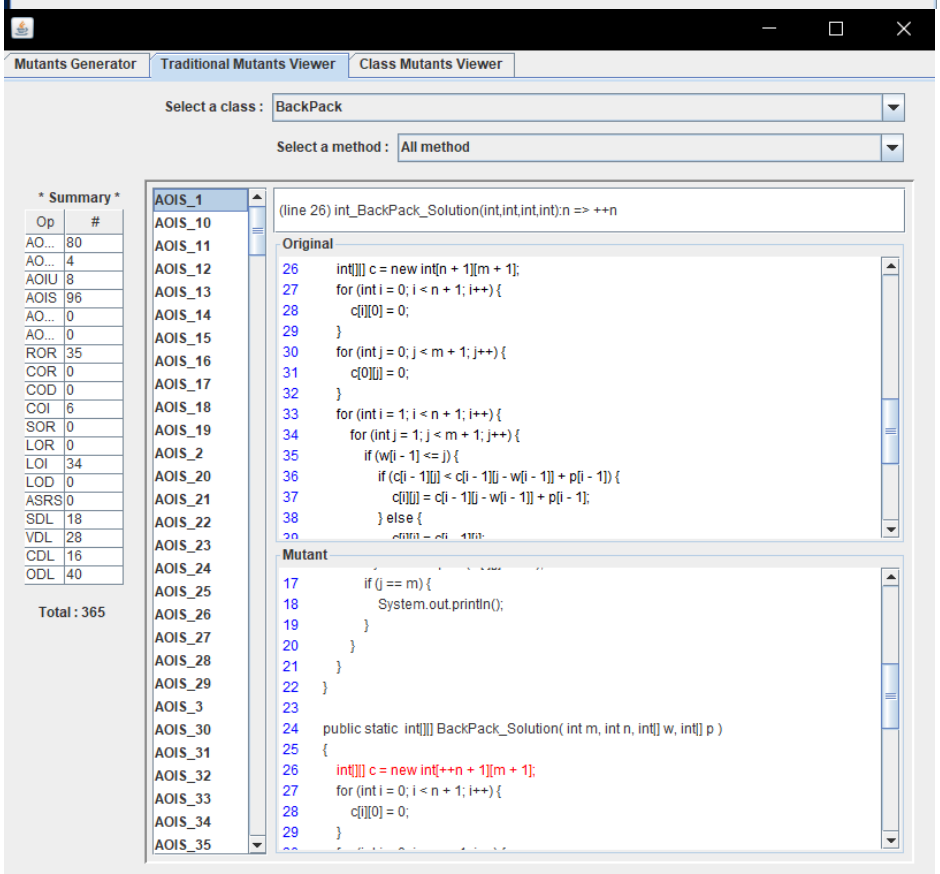
Run the “GenMutants.cmd” file enter the GUI interface of MuJava



Then choose those two java files and the mutant operators, by clicking the “generate” button to produce the mutants which we can

see below:





The mutant part was shown in red.

To test the mutant programs, I wrote two java files one for testing the implementation of BubbleSort the other for testing the implementation of BackPack both use junit. The source code as shown below:

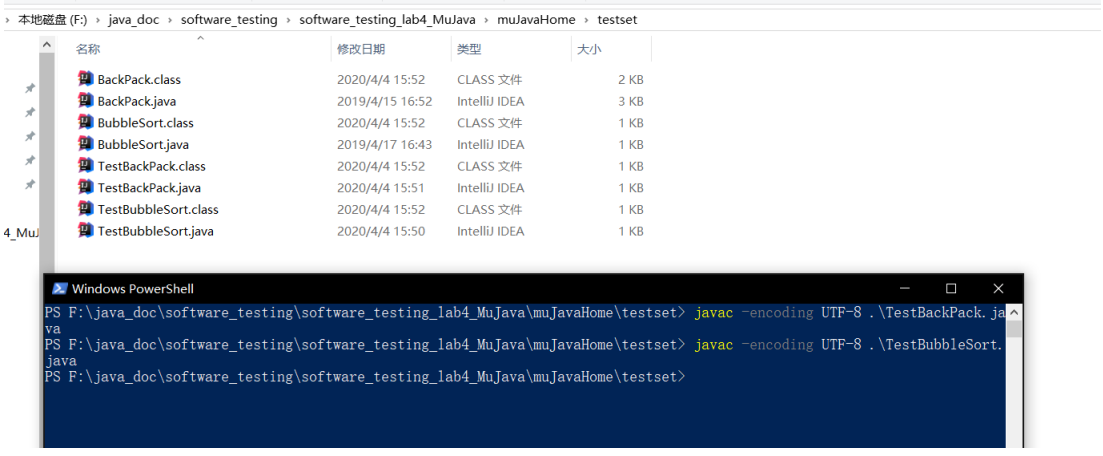
TestBubbleSort.java

```java
import static org.junit.Assert.assertArrayEquals;
import org.junit.Test;

public class TestBubbleSort {
    @Test
    public void test(){
        int array[] = new int[]{1,6,2,2,5};
        int result[] = new int[]{1,2,2,5,6};
        assertArrayEquals(result,BubbleSort.BubbleSort(array));
    }
}
```

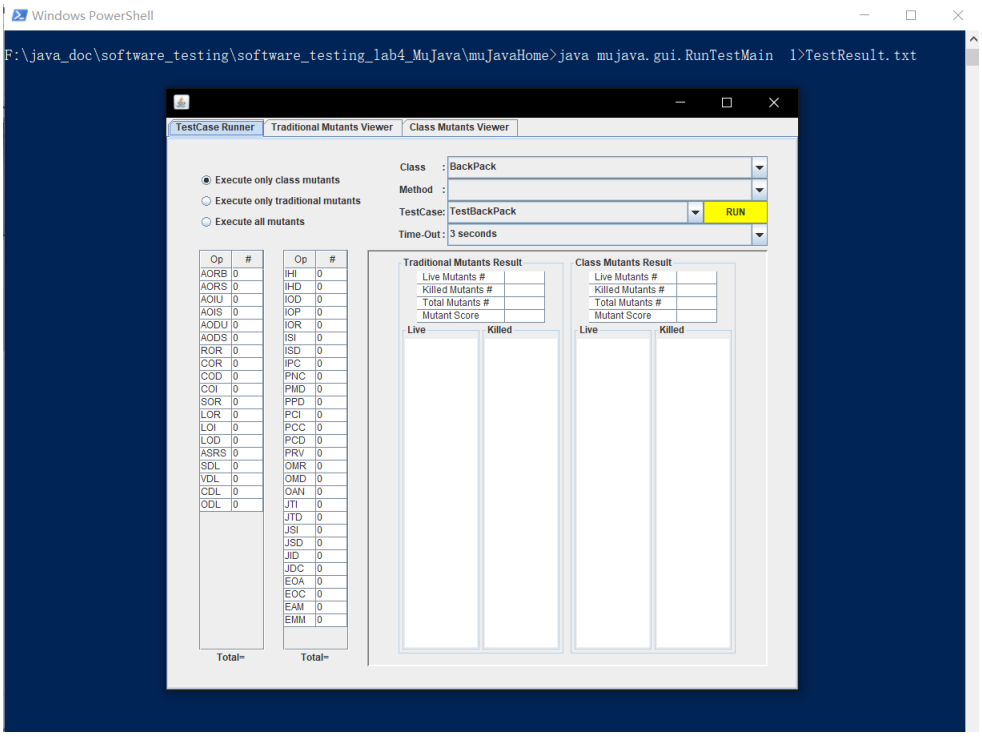
TestBackPack.java

```java
import static org.junit.Assert.assertArrayEquals;

import org.junit.Test;

public class TestBackPack {
    @Test
    public void test() {
        int m = 10;
        int n = 2;
        int w[] = {3, 4, 5};
        int p[] = {4, 5, 6};
        int[][] result = {{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
                {0, 0, 0, 4, 4, 4, 4, 4, 4, 4, 4},
                {0, 0, 0, 4, 5, 5, 5, 9, 9, 9, 9},
                {0, 0, 0, 4, 5, 6, 6, 9, 10, 11, 11}};
        assertArrayEquals(result, BackPack.BackPack_Solution(m, n, w, p));
    }

}
```

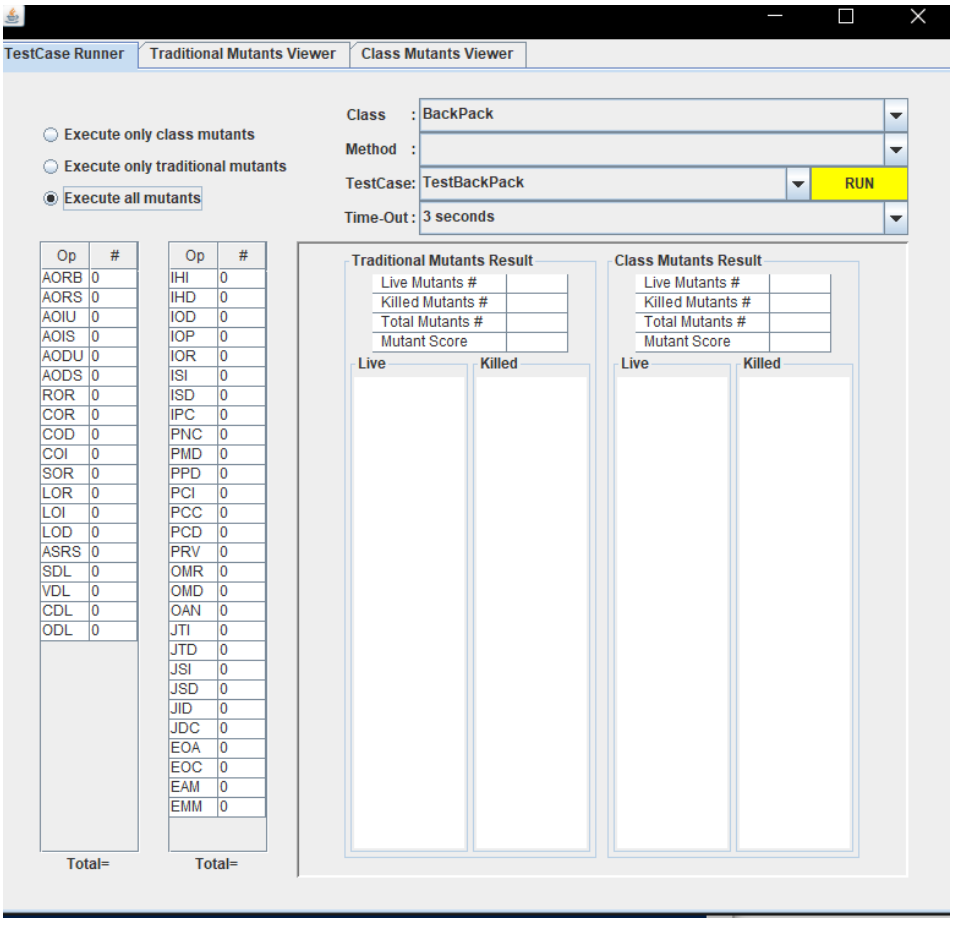
Put those two testing scripts mentioned above and BubbleSort.java, BackPack.java into the folder “testset” and use “javac” to generate the .class files.



In the end, use the RunTest.cmd to execute the testing process.

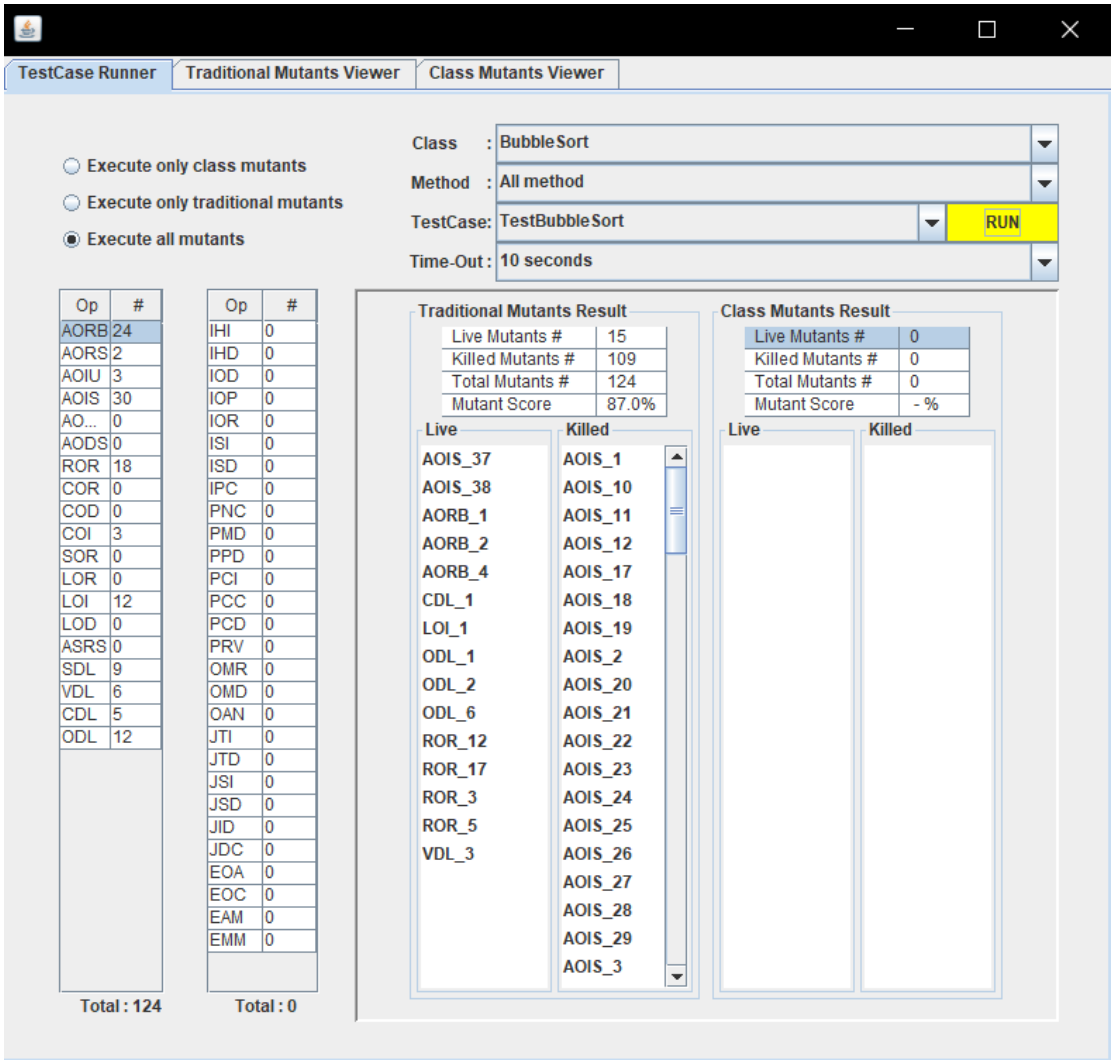


Choose the option, for example I choose to test all mutants, and click the "run" button to start the test.

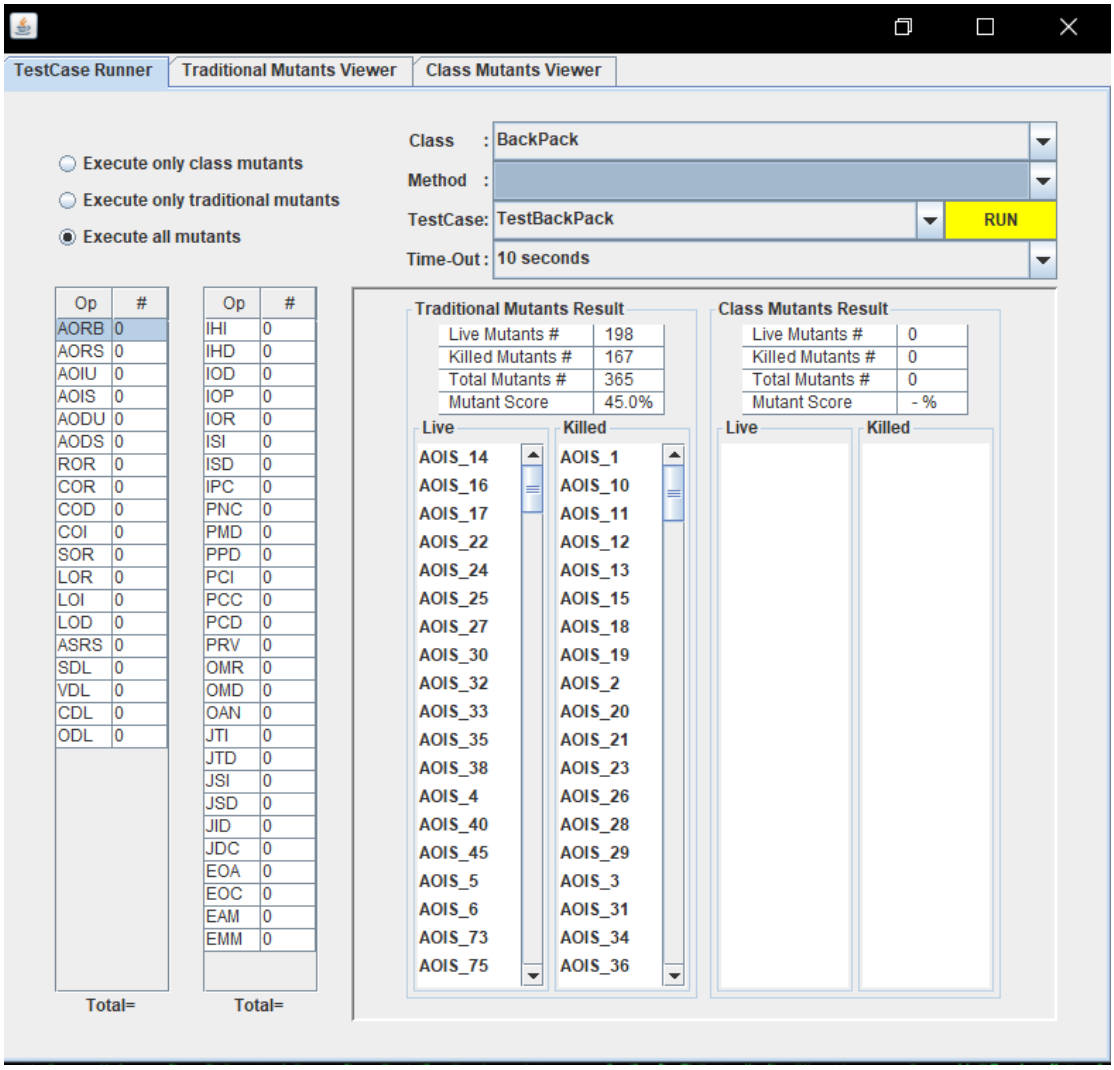


3、Experiment Result

The result of mutant test of Bubble Sort are shown below, from which we can tell that there are 124 mutants in total, and 109 of them were killed, 15 of them were left and the mutant score is 87% which is a somehow nice score.



The result of mutant test of Back Pack are shown below, from which we can tell that there are 365 mutants in total, and 167 of them were killed, 198 of them were left and the mutant score is 45% which, by contrast, is not good enough and still need to be improved.



In the end, I opened the TestResult.txt as well, from which we can see the related detailed about the test process including expected value, actual value, etc.