****

**软件测试技术第四次实验报告**



**学 院 软件学院**

**专 业 软件工程**

**年 级 16级**

**姓 名 李今**

**2019年 4月 24 日**

# 软件测试技术第四次实验报告

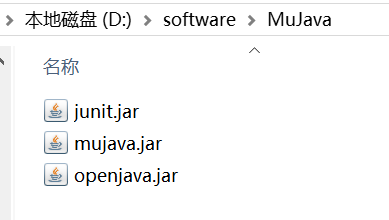
git URL

https://github.com/riginAI/test/tree/master/Lab4

1. 需求分析
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.
4. 环境配置

安装MuJava

（1）先把从老师所给链接下载的MuJava解压放在任意目录下



（2）之后添加环境变量

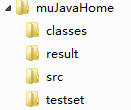
D:\software\mujava\mujava.jar;

D:\software\mujava\openjava.jar;

D:\software \mujava\junit.jar

（3）新建muJavaHome目录，并在目录下新建mujava.config 文件，内容为：MuJava\_HOME=C:\mujavaHome

之后在muJavaHome目录下创建如下结构，方便之后存放文件

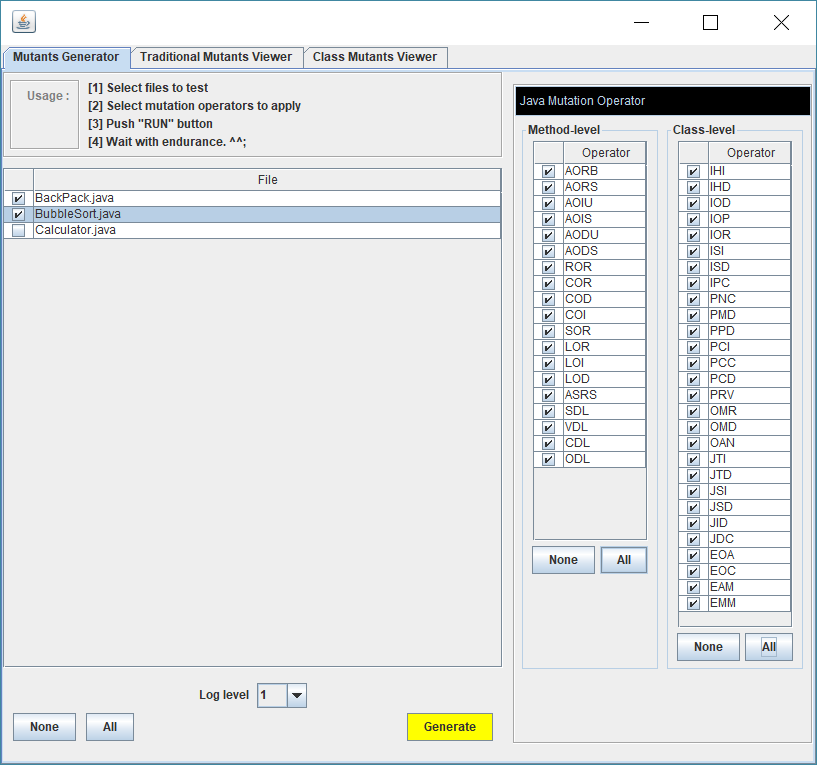


（4）在C:\mujavaHome下创建两个Txt，另存为cmd后缀，方便运行mujava程序。txt分别内容为：

GenMutants.cmd: java mujava.gui.GenMutantsMain

RunTest.cmd: java mujava.gui.RunTestMain > TestResult.txt

（5）最后将backpack.java和bubblesort.java复制到src文件夹，将通过Eclipse编译生成的.class文件复制到classes文件夹，运行GenMutants.cmd。在mutants界面勾选生成变异体的选项，点击generate即可开始生成变异体



1. junit测试代码

为了对生成的变异体进行测试，我们要先准备好针对backpack和bubblesort的单元测试。本人的测试代码相对简单。

BackPackText.java

@Test

**public** **void** test() {

**int** m = 10;

**int** n = 3;

**int** w[] = {3, 4, 5};

**int** p[] = {4, 5, 6};

**int** c[][] = backpack.*BackPack\_Solution*(m, n, w, p);

**int** result[][] = {{0,0,4,4,4,4,4,4,4,4},

{0,0,4,5,5,5,9,9,9,9},

{0,0,4,5,6,6,9,10,11,11}};

**for** (**int** i = 1; i <= n; i++) {

**for** (**int** j = 1; j <= m; j++) {

*assertEquals*(c[i][j],result[i-1][j-1]);

}

}

}

BubbleSortTest.java

@Test

**public** **void** test1() {

**int** arr1[] = **new** **int**[]{1,6,2,2,5};

BubbleSort.*BubbleSort*(arr1);

**int** arr2[] = **new** **int**[]{1,2,2,5,6};

*assertEquals*(Arrays.*toString*(arr1),Arrays.*toString*(arr2));

}

@Test

**public** **void** test2() {

**int** arr1[] = **new** **int**[]{1,0,6,2,2,5};

BubbleSort.*BubbleSort*(arr1);

**int** arr2[] = **new** **int**[]{0,1,2,2,5,6};

*assertEquals*(Arrays.*toString*(arr1),Arrays.*toString*(arr2));

}

@Test

**public** **void** test3() {

**int** arr1[] = **new** **int**[]{2,1,6,2,2,5};

BubbleSort.*BubbleSort*(arr1);

**int** arr2[] = **new** **int**[]{1,2,2,2,5,6};

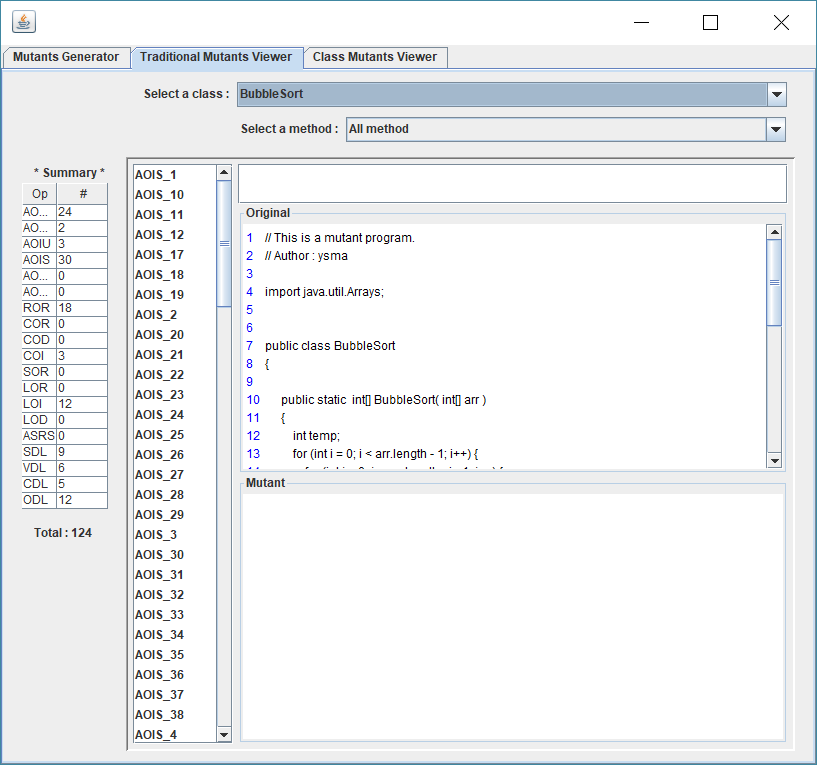
*assertEquals*(Arrays.*toString*(arr1),Arrays.*toString*(arr2));

}

1. 实验结果

（1）生成mutants

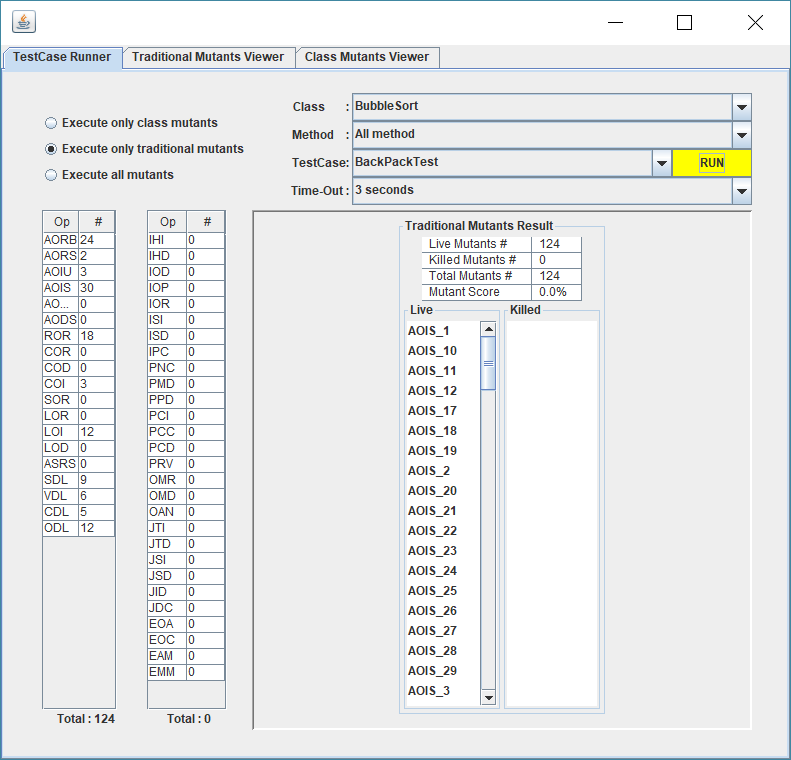
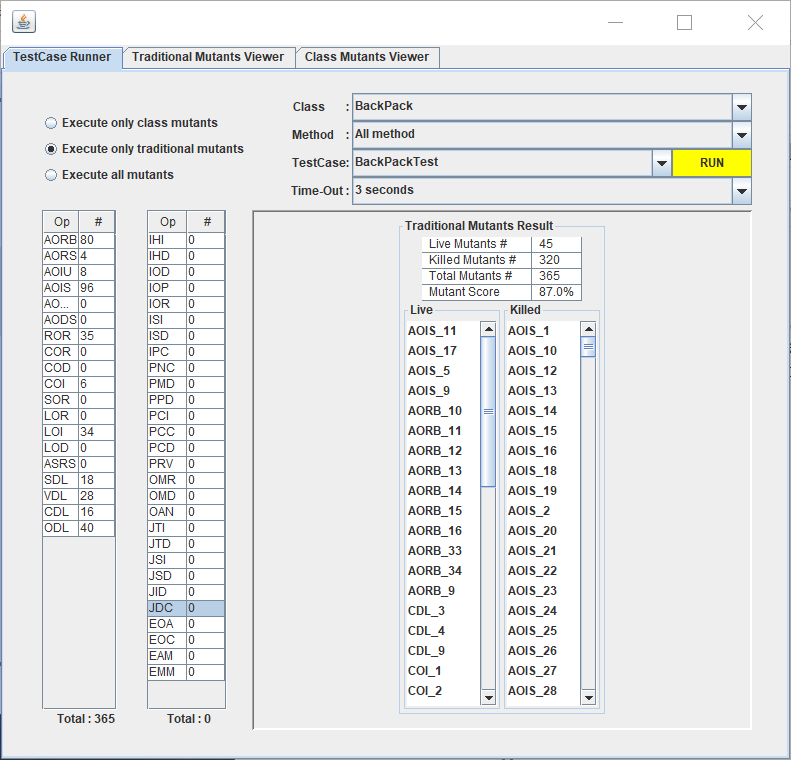
点击generate可以在traditional viewer上看到生成的各种变异体



变体.java与.class位于result文件夹。

（2）运行变异测试

将JUnit测试用例的.class文件复制在testset文件夹中，运行RunTest.cmd即可开始变异体的测试。



1. 总结

这次实验内容对我来说是十分新奇的，因为之前在学习mutants时只是接触了变异体这一概念，而落实到实践中时发现借助mutant工具，测试变异体还是非常方便的。而且这次实验还和之前的junit单元测试结合了起来，让我在熟悉单元测试的同时，对两者的配合也更加得心应手。