****

**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 大三**

**姓 名 张聪**

**2019年 4 月 24 日**

# 软件测试技术第一次实验报告

git网址链接：https://github.com/likeAnEgg/SoftwareTest/tree/master/lab

1. 需求分析

安装mujava。已知两个小程序，BubbleSort.java是bubble sort算法的一个实现，BackPack.java是01背包问题的一个解决方案。尝试用mujava生成2个给定程序的突变体。

1. 概要设计

先安装并配置好mujava，使用mujava生成两个所给java文件的变异体.在eclipse中编写junit测试文件,完善后放入mujava目录中,用mujava进行变异测试.

注意到所给的两个java文件中包含有main函数,main函调用了所属类中的全部其他函数,因此检查main函数就可以覆盖全部代码。junit可以直接将main函数当做普通函数调用,校验正确性就依靠各自main函数的控制台输出即可。通过把标准输出定向至ByteArrayOutputStream中去得到输出的字符串,然后可以对其内容进行断言。

1. 详细设计

（一）设计BubbleSort.java 的测试类TestBS.java

**import** java.io.ByteArrayOutputStream;

**import** java.io.PrintStream;

**import** java.util.Arrays;

**import** org.junit.Assert;

**import** org.junit.Test;

**public** **class** TestBS {

@Test

**public** **void** testBS(){

BubbleSort bs=**new** BubbleSort();

**final** ByteArrayOutputStream outContent = **new** ByteArrayOutputStream();

System.*setOut*(**new** PrintStream(outContent));

bs.*main*(**null**);

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

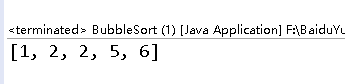
String os=Arrays.*toString*(rt)+"\r\n";

Assert.*assertEquals*(os,outContent.toString());

}

}

注：BubbleSort类输出如图



编写完成并测试通过后，将TestBS.java和TestBS.class放入testset文件夹下

（二）BackPack.java的测试类TestBP.java

**import** java.io.ByteArrayOutputStream;

**import** java.io.PrintStream;

**import** org.junit.Assert;

**import** org.junit.Test;

**public** **class** TestBP {

@Test

**public** **void** testBP(){

BackPack bp=**new** BackPack();

**final** ByteArrayOutputStream outContent = **new** ByteArrayOutputStream();

System.*setOut*(**new** PrintStream(outContent));

bp.*main*(**null**);

String os="0"+"\t"+"0"+"\t"+"4"+"\t"+"4"+"\t"+"4"+"\t"+"4"+"\t"+"4"+"\t"+"4"+"\t"+"4"+"\t"+"4"+"\t"+"\r\n"

+"0"+"\t"+"0"+"\t"+"4"+"\t"+"5"+"\t"+"5"+"\t"+"5"+"\t"+"9"+"\t"+"9"+"\t"+"9"+"\t"+"9"+"\t"+"\r\n"

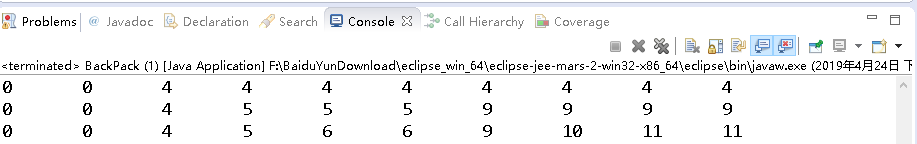
+"0"+"\t"+"0"+"\t"+"4"+"\t"+"5"+"\t"+"6"+"\t"+"6"+"\t"+"9"+"\t"+"10"+"\t"+"11"+"\t"+"11"+"\t"+"\r\n";

Assert.*assertEquals*(os,outContent.toString());

}

}

注：BackPack类输出如图

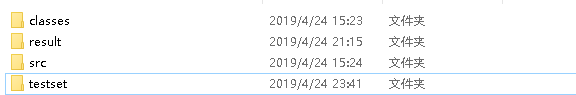


编写完成并测试通过后，将TestBP.java和TestBP.class放入testset文件夹下

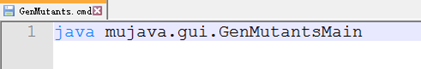
（三）配置mujava

下载老师提供的压缩包，解压到E盘，添加classpath变量E:\mujava\mujava.jar;E:\mujava\openjava.jar;E:\mujava\junit.jar。

新建文件夹muJavaHome，将mujava.config放至该目录下，并手动创建如下目录结构



新建GenMutants.cmd和RunTest.cmd用于运行变体产生和运行测试的命令。

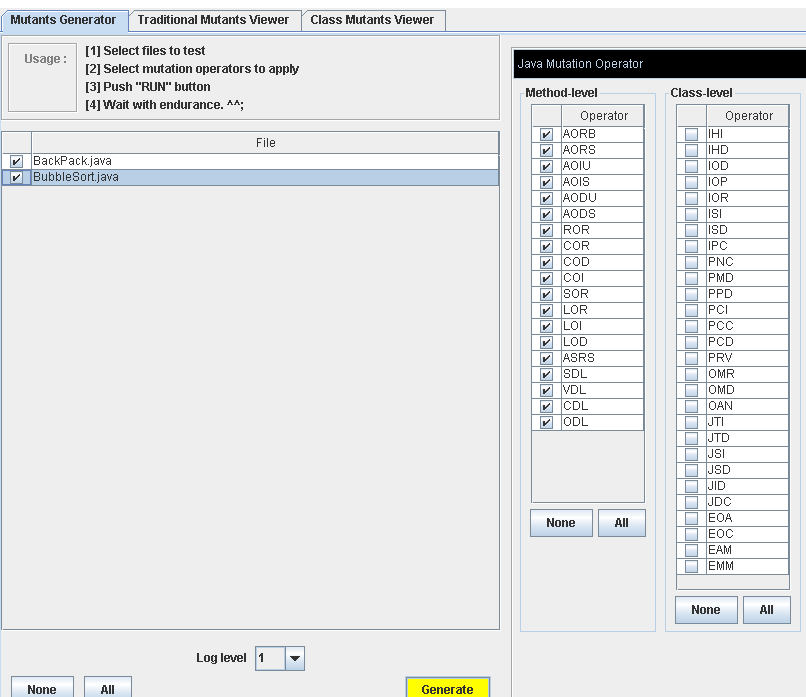


IMG_257

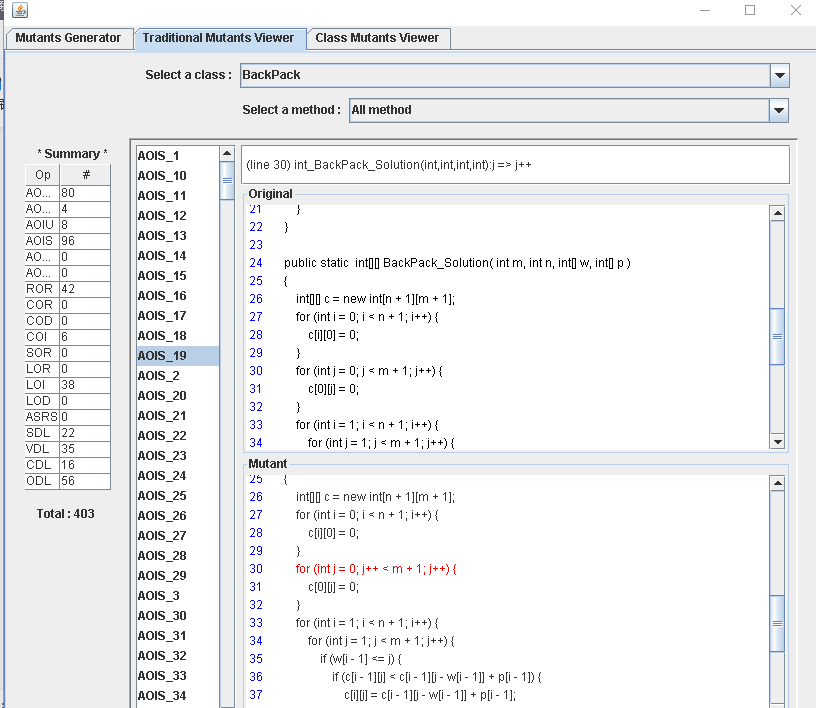
（四）生成变异体

将要求中给出的java文件编译成class文件，将java文件复制到src目录下，class文件复制到classes目录下，运行GenMutants.cmd。

在弹出的图形界面中选中想用的变异操作符，点击generate生成变异体

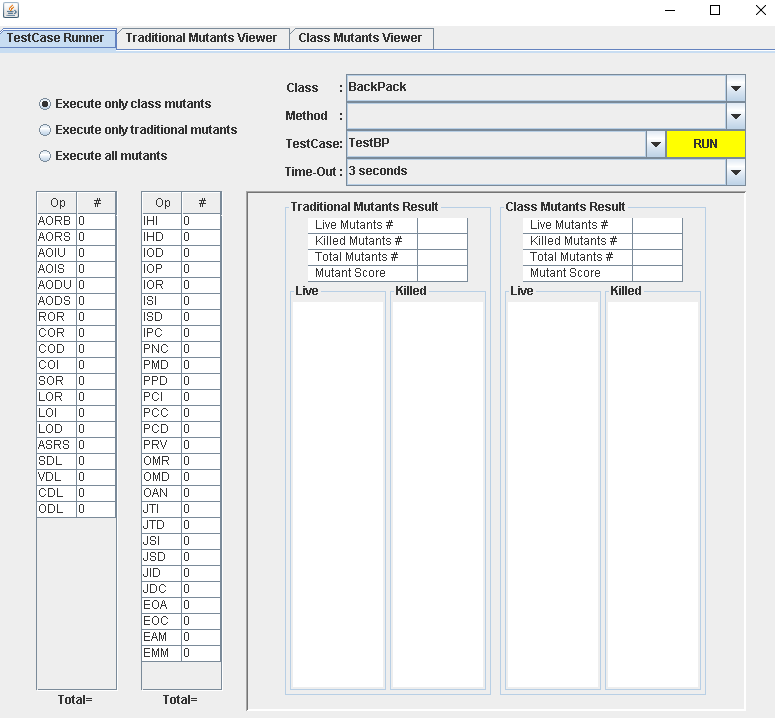


在viewer中可以浏览生成的变异体



（五）进行变异测试

将编写的junit测试代码java文件以及编译生成的class文件复制到testset文件下，运行RunTest.cmd，打开进行测试的图形界面

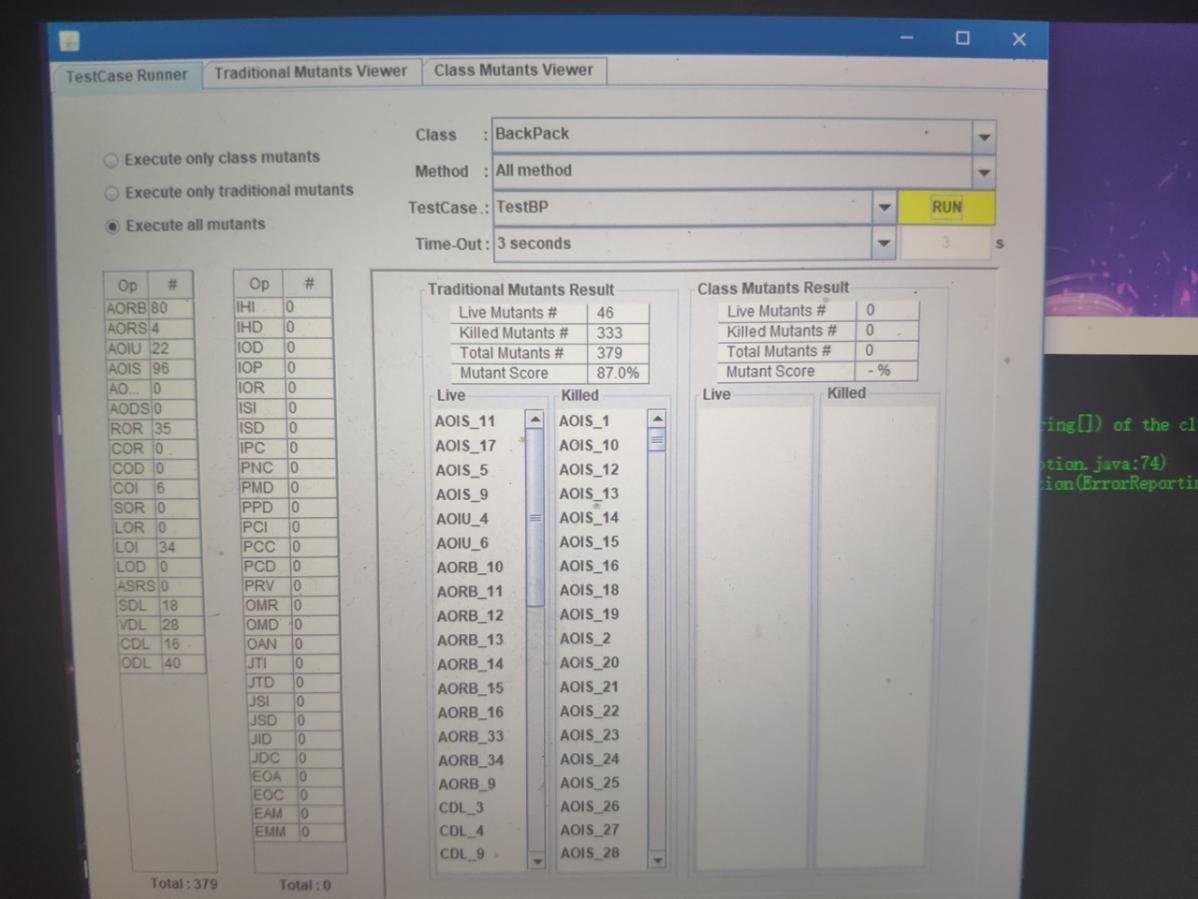


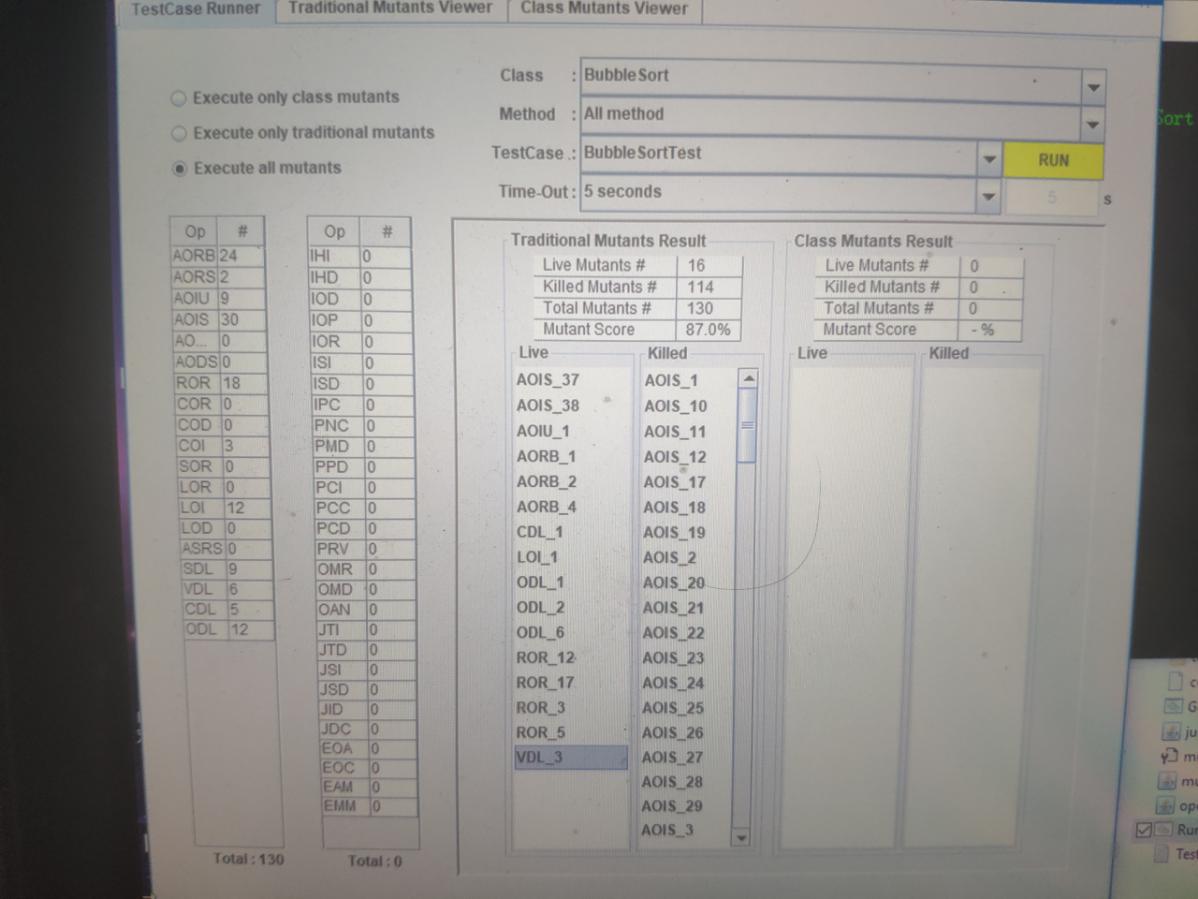
结果详见后面的“测试结果”

1. 调试分析

一开始mujava出现各种报错，我将jdk版本改成1.7后可以生成变异体，但变异测试时又发生无法杀死变异体的问题。最后将jdk版本改成1.8重新配置并生成变异体后得到解决。

1. 测试结果





1. 总结

经过本次实验,我初步了解了如何使用mujava进行变异测试,并认识到这些工具对软件开发中的测试活动带来的便利.对软件测试流程有了更深入的了解和掌握,且对课堂上老师讲的变异测试等知识点有了更深刻的了解,受益匪浅。