单元测试 JUnit 实验指导手册

通过本次实验了解通过 JUnit 进行黑盒测试的基本方法。

1 下载实验相关环境

单元测试实验材料 ftp 地址：ftp://10.21.11.21/教师课件/软件工程系/王潇杨/软件质量与测试/lab3/

其中：

- JUnit 单元测试实例源代码：junitfirst_src.rar（必选）
- JUnit 软件包：junit4.4.zip（必选）
- Jdk 编译器：jdk-7u25-windows-i586 （当实验室机器 jdk 不兼容或者出现其他问题时可以使用）
- 集成开发环境：eclipse_helios_windows.rar（当实验室机器 eclipse 不兼容或者出现其他问题时可以使用）
- 深入学习的网址：http://blog.csdn.net/xifeijian/article/category/1377693（对本次实验涉及的语法和函数有详细的解释，可以进行参照学习）
- Unit Testing in JUnit.pdf：可以用于课后对 JUnit 进行进一步学习。

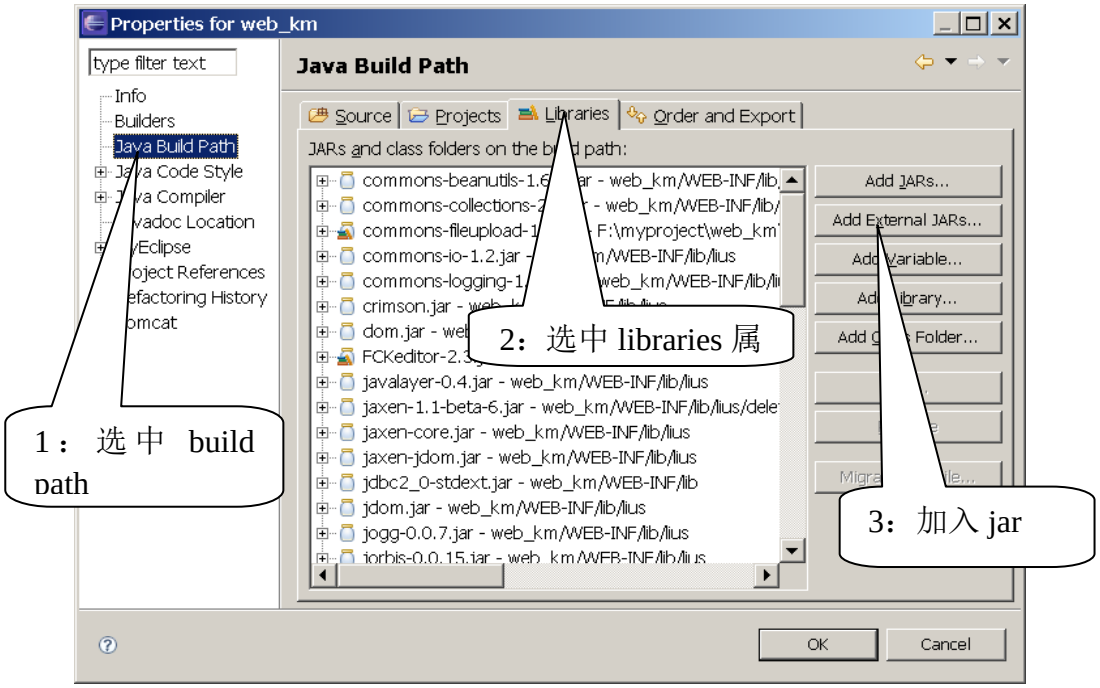
2 单元测试实例操作步骤

(1) JUnit 安装配置与简单测试类编写

1) 将 junit4.4.zip 解压至 D 盘 junit4.4 目录下，观察 jUnit 所包含的主要的类库文件：junit-4.4.jar。

2) 将 junitfirst_src.rar 解压至 D 盘 junitfirst_src 目录中。

3) 启动 eclipse，新建一个 java 项目，名为 junitfirst2018，选中该项目，点菜单 project->properties,弹出一个对话框。点击 Java Build Path，在右边出现的属性页中选中 Libraries。再点击右边的 Add External Jars，并选择步骤 1 中 D:\junit4.4 目录下的 junit-4.4.jar 文件。



4) 将步骤 2 中得到的所有源文件复制至项目 junitfirst2018 所在的目录中

5) 刷新项目 junitfirst2018，在(default package)下面找到这些源代码文件。

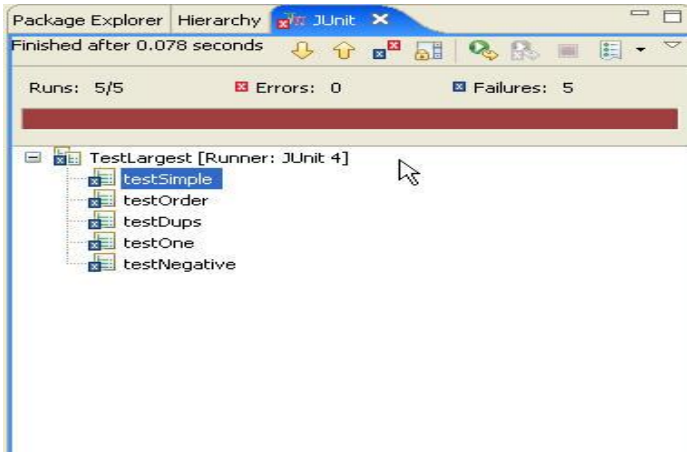
6) 双击 Largest.java 文件，打开浏览其源代码，该 java 程序完成一个寻找一个整数列表中最大数的功能。请你通过代码走读方式查找其中的错误，并进行记录。

7) 双击 TestLargest.java 文件，打开浏览其源代码，该 java 程序为经典的基于 junit 测试框架写的测试用例。

基于 Junit 框架的测试类程序编写简要指南

- 必须导入 junit.framework.*类库包
- 所写的 java 类必须继承 TestCase 类
- 其构造函数使用字符串作为参数，同时调用父类构造函数
- 编写测试用例集函数，其函数必须使用以 test 开头的函数名，如在 TestLargest.java 中写了 4 个测试用例集，分别名为 testSimple, testOrder, testDups,testone 和 testNegative，它们分别进行了简单用例测试，极端有序用例的测试，有重复值用例的测试，唯一值用例的测试和负值用例的测试。
- 在每个测试用例集函数内，junit 使用断言函数判断预期值与实际值进行的比较结果，比如 assertEquals 用来做等值比较，若相等，表示该用例通过，否则表示失败。
- 上述带有 test 前缀的测试用例集函数会在 junit 框架的驱动下，全部进行自动运行，并返回测试结果。

8) 鼠标右键单击 TestLargest.java 文件，选择 Run AS，选择 Junit Test，在 junit 框架的驱动下，运行 TestLargest 类。运行结果如图。如图所示，5 个用例集都没有运行通过。表明 Largest 类有错误。



9) 修改 Largest 类的错误，MAX_VALUE 改为 MIN_VALUE，index=1 改为 index=0，index<List.length-1 改为 index<=List.length-1

10) 改正错误后再次执行 TestLargest，直至通过全部测试用例。

(2) 选择特定的几个测试用例集进行测试

1) 有时我们不需要将所有的测试用例集函数都自动运行，我们需要选择几个用例集函数来运行，这就需要使用 TestLargestGroup.java 里的功能了。

2) 打开 TestLargestGroup.java，关注以下程序段

```
public static Test suite(){
        TestSuite suite = new TestSuite();
        suite.addTest(new TestLargest("testOrder"));
        suite.addTest(new TestLargest("testDups"));
        suite.addTest(new TestLargest("testNegative"));
        return suite;
}
```

3) 建立返回 Test 类型的测试组函数 suite，在其内部新建 TestSuite 对象 suite，挑选需要运行的测试用例集，并通过 suite.addTest(new TestLargest(“测试用例集函数名”))将它们添加到测试用例组中去。

4) 使用同样方法运行 TestLargestGroup.java，测试特定的几个测试用例集函数，观察所执行的结果。

(3) 同时进行多个测试类的测试执行

1) 再写一个测试类程序如 Testadd.java 所示。在 Testadd 测试类中，建立了 TestAddition，TestSubtraction 两个测试用例集。

2) 5双击 TestClassComposite.java 源程序，见如下代码：

```
static public Test suite(){
        TestSuite suite = new TestSuite();
        suite.addTestSuite(Testadd.class);
        suite.addTest(TestLargestGroup.suite());
        return suite;
  }
```

3) 在新建的测试组对象 suite 中通过两种方法添加另一个测试组对象 suite.addTestSuite(Testadd.class)和 suite.addTest(TestLargestGroup.suite())

4) 使用同样方法运行 TestClassComposite.java，测试多个测试类的几个测试用例集函数，观察所执行的结果。

(4) 脱离 eclipse junit plugin 的支持，独立运行测试类

1) 运行 TestRuner.java 源程序，通过注释分析各个模块代码。

3 实验目的

体验测试驱动的开发：在没有写某个功能的代码之前，先把能测试出这个功能是否正常的测试用例设计出来，然后把测试用例用 Junit 实现为测试类，最后才是编写实现某功能的代码。要验证功能代码是否正确，只需要执行测试类即可。这就是测试驱动的开发（先写测试用例，后写实际的代码）。

4 本次实验内容

1) 假设你受客户委托开发一个具有一定功能的 Java 类，类名为 OnefunCls，请完成该类的开发。功能如下：

   a) 编写一个 java 类，实现静态方法 int GetMiddle(int [] List)，该方法取得在 list 中最接近均值的元素，如果两个元素距离均值距离相同，则返回较大的元素。如[3,5,2,5,1]，返回 3；

   b) 实现静态方法 String D2H(int d),该方法将十进制数 d 转换为十六进制数，其转换结果形式为 100FH(最后一个字符 H 表示十六进制)。如 d=124 时，返回 7cH;

2) 假设你作为用户对该产品进行质量检验，请使用等价类划分和边界值分析方法进行测试用例的设计，并基于 JUnit 编写测试代码，用以检验这个产品的质量。注：在代码编写时请添加充足注释，对代码进行解释，并说明测试用例针对等价类划分和边界值分析方法的实现情况。

3) 本次实验每人独立完成，请将用 eclipse 编写的 Junit 单元测试项目的整个文件夹，包括 JAVA 类(客户功能实现源代码)和 Test 类，一起打成压缩包。文件名：班号_姓名_学号_unittest。提交地址：ftp://10.21.11.21/学生作业上传/王潇杨/软件质量与测试/lab3/