**《软件测试》**

**实验报告二 ——边界值分析方法**

**姓 名： 黄文轩 学 号：2019176203**

**院 系： 计算机与信息学院 专 业： 数据科学与大数据技术**

**实 验 室： 实验日期：2022/3/14**

**总评成绩： 审阅教师：**

目录

[**《软件测试》** 1](#_Toc98170011)

[**实验报告一 ——JUnit测试方法** 1](#_Toc98170012)

[一、实验目的： 3](#_Toc98170013)

[二、实验环境： 3](#_Toc98170014)

[三、实验要求： 3](#_Toc98170015)

[四、实验步骤与内容 3](#_Toc98170016)

[五、结论分析与体会 6](#_Toc98170017)

[六、仓库地址 6](#_Toc98170018)

[附录 7](#_Toc98170019)

### 一、实验目的：

1. 练习基本的编程能力；
2. 学习JUnit自动测试框架，掌握非参数化和参数化测试方法；
3. 理解以下三个题目，这是后面讲解黑盒测试方法和白盒测试方法的基础案例；

### 二、实验环境：

Eclipse2020,JUnit

### 三、实验要求：

1. 使用Java语言实现以下三个程序；
2. 使用测试用例验证程序的正确性（JUnit）；
3. 测试用例数据可以参考naiveX.xls，并运行naiveX.xls;(p24页第8题）；

### 四、实验步骤与内容

1. **题目：三角形题目**

**输入**3个整数a、b和c分别作为三角形的三条边，要求a、b和c必须满足以下条件：

1、整数

2、3个数

3、边长大于等于1，小于等于100

4、任意两边之和大于第三边

**输出**为5种情况之一：

如果不满足条件1、2、3，则程序输出为“输入错误”。

如果不满足条件4，则程序输出为“非三角形”。

如果三条边相等，则程序输出为“等边三角形”。

如果恰好有两条边相等，则程序输出为“等腰三角形”。

如果三条边都不相等，则程序输出为“一般三角形”。

1. **测试用例：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 边长a | 边长b | 边长c | 预期输出 | 实际输出 |
| 1 | 50 | 50 | 1 | 等腰三角形 | 等腰三角形 |
| 2 | 50 | 50 | 2 | 等腰三角形 | 等腰三角形 |
| 3 | 50 | 50 | 50 | 等边三角形 | 等边三角形 |
| 4 | 50 | 50 | 99 | 等腰三角形 | 等腰三角形 |
| 5 | 50 | 50 | 100 | 非三角形 | 非三角形 |

…

…

…

…

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 490 | 100 | 50 | 1 | 非三角形 | 非三角形 |
| 491 | 100 | 50 | 2 | 非三角形 | 非三角形 |
| 492 | 100 | 50 | 50 | 非三角形 | 非三角形 |
| 493 | 100 | 50 | 99 | 不等边三角形 | 不等边三角形 |
| 494 | 100 | 50 | 100 | 等腰三角形 | 等腰三角形 |
| 495 | 100 | 99 | 1 | 非三角形 | 非三角形 |
| 496 | 100 | 99 | 2 | 不等边三角形 | 不等边三角形 |
| 497 | 100 | 99 | 50 | 不等边三角形 | 不等边三角形 |
| 498 | 100 | 99 | 99 | 等腰三角形 | 等腰三角形 |
| 499 | 100 | 99 | 100 | 等腰三角形 | 等腰三角形 |
| 500 | 100 | 100 | 1 | 等腰三角形 | 等腰三角形 |

1. **代码实现：**

|  |
| --- |
| public class Triangle {  public String classify(int a, int b, int c) {  if (a <1 || a>100 || b<1 || b>100 || c<1 || c> 100) {  return "输入错误";  }  if (!((a + b > c) && (a + c > b) && (b + c > a))) {  return "非三角形";  } else if (a == b && a == c && b == c) {  return "等边三角形";  } else if (a != b && a != c && b != c) {  return "不等边三角形";  } else {  return "等腰三角形";  }  }  }  测试代码：  import org.junit.jupiter.api.DisplayName;  import org.junit.jupiter.api.Test;  import static org.junit.jupiter.api.Assertions.\*;  class Test1Test {  @Test  @DisplayName("输入错误")  void parameters\_error\_test() {  Triangle triangle = new Triangle();  String type = triangle.classify(0, 4, 5);  assertEquals("输入错误", type);  }  @Test  @DisplayName("不等边三角形")  void scalene\_test() {  Triangle triangle = new Triangle();  String type = triangle.classify(3, 4, 6);  assertEquals("不等边三角形", type);  }  @Test  @DisplayName("非三角形")  void not\_triangle\_test() {  Triangle triangle = new Triangle();  String type = triangle.classify(3, 1, 1);  assertEquals(" ", type);  }  }  测试结果：    覆盖率： |

1. **结论分析与体会**

学会了软件测试边界值分析方法。

1. **仓库地址**

https://github.com/Osilly/Software-Testing-Experiment