|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | **学生学号** | 0122110880204 | **实验课成绩** |  | |
|  |
| 学 生 实 验 报 告 书 |
| |  |  |  | | --- | --- | --- | | 课程名称 |  | 软件质量保证与测试B | | 开课学院 |  | 计算机与人工智能学院 | | 指导老师姓名 |  | 胡文华 | | 学生姓名 |  | 宋文博 | | 学生班级 |  | 软件2101 | |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | 2023 | —— | 2024 | 学年第 | 一 | 学期 | |

实验课程名称：软件质量保证与测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 实验项目名称 | 黑盒测试（Black-box testing） | | | 实验成绩 |  |
| 实验者姓名 | 宋文博 | 专业班级 | 软件2101 | 组别 | / |
| 同组者 | / | | | 实验日期 | 2023年11月20日 |
| 1. 实验分析与设计    1. 实验目的   (1) 掌握黑盒测试分析和生成测试用例的方法；  (2) 掌握边界值分析方法产生测试集。   * 1. 实验内容   **2.1** **问题描述：三角形问题**  在一个程序中，依次输入三个整数 a，b，c 作为三角形的三条边。三边不同的关系，输出三角形的类型：Equilateral（等边的）、Isosceles（等腰的）、Scalene（三边不等的三角形)或者 Not A Triangle（非三角形）。这里暂时不考虑直角三角形。  判断过程的流程图如图所示。    **2.2 问题描述的改进**  基于前述的三角形问题，继续增加如下限制，量化问题描述进行改进。  C1: 1 ≤ a ≤ 200  C2: 1 ≤ b ≤ 200  C3: 1 ≤ c ≤ 200 C4: a < b + c  C5: b < a + c C6: c < a + b  如果输入整数值范围超出 C1, C2, C3指定，则输出提示信息“Value of b is not in the range of permitted values.”（以 C2 不满足为例，其它类似）。  当输入a,b,c满足 C4, C5, C6指定后，再判断三角形类型（Equilateral / Isosceles / Scalene/ Not A Triangle）。  判断过程三角形的流程图如图所示。    **3.1 编程任务**  根据图 1 和图 2 描述的流程图描述，分别进行编程实现。  **3.2 边界值分析测试实验**  通过边界值分析方法生成测试集，不断地修改程序，直到所有测试用例集测试均通过。 测试用例集的生成分别基于普通边界值（normal boundary value）测试和最坏情形边界值（worst-case boundary value）测试两种情况进行。例如，现已给出如下普通边界值测试用例表 1。  表 1 普通边界值测试用例  继续设计最坏情形用例边界值，并将两种测试集实现，用于程序的测试。   * 1. 主要仪器设备及耗材   (1) PC 机；  (2) 编程语言：Java；  (3) 开发环境：vscode； | | | | | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. 实验调试与结果分析    1. 实验设计     从题目给出的流程图我们可以发现其中有明显的错误。图中有明显错误（14，17，19判断分支错误），根据三角形两边之和大于第三边的性质，上述分支Y、N颠倒。  根据修改后的流程图我们可以写出代码（见“实验代码”）。    而在上述流程图中，红色框中流程不符合常规逻辑，尽管不会出错但会造成代码冗余和无效运行导致代码可读性差以及性能降低。当有一个数不在范围内时就应该退出程序进行下一次输入的读取，而不再判断其他边。  随后根据流程图形成若干函数  is\_value\_valid判断是否在范围内  is\_triangle根据三角形性质判断三边是否合法  what\_type\_triangle2改进后的判断三角形类型函数  根据给定的条件，以下是最坏情况下的边界值测试用例的设计：  最小边界值测试用例  最大边界值测试用例  边界值测试用例 - 边界情况  边界值测试用例 - 非边界情况  失败测试用例 - 违反条件 C4  失败测试用例 - 违反条件 C5  失败测试用例 - 违反条件 C6  a b c expected Output   |  |  |  |  | | --- | --- | --- | --- | | 1 | 1 | 1 | Equilateral | | 200 | 200 | 200 | Equilateral | | 1 | 200 | 200 | Isosceles | | 200 | 1 | 200 | Isosceles | | 200 | 200 | 1 | Isosceles | | 2 | 199 | 199 | Isosceles | | 199 | 2 | 199 | Isosceles | | 199 | 199 | 2 | Isosceles | | 1 | 1 | 199 | Not a triangle | | 1 | 199 | 1 | Not a triangle | | 199 | 1 | 1 | Not a triangle | | 1 | 199 | 199 | Isosceles | | 199 | 1 | 199 | Isosceles | | 199 | 199 | 1 | Isosceles | | 199 | 199 | 201 | c invalid | | 199 | 201 | 199 | b invalid | | 201 | 199 | 199 | a invalid |   再结合题目中给出的用例，我将所有测试用例以及预期输出导入一张表格中，形成csv文件，再使用程序读取，将每一行测试用例作为参数输入程序，将预期输出与程序实际输出进行对比。实现批量用例的测试。   * 1. 实验代码   根据流程图1修改后编写的代码：    根据流程图2修改后编写的核心代码：    实现批量测试程序：       * 1. 实验结果截图   由于程序正确时，测试用例输入测试程序不会有任何输入，所以我修改了测试用例以显示若程序输出不符合预期的输出，**注意：我将错误的测试用例输入程序**。  左图是正确的测试用例  接下来我将修改测试用例的预期输出，修改为错误的以显示测试程序输出：    上图是输出结果。（再次提醒，我用错误的测试用例测试程序，所以此处实际输出才是正确的）若测试用例正确，测试程序将不会有任何输出。   1. 实验小结、建议及体会   测试是软件开发过程中至关重要的一环。通过设计和执行测试用例，我们可以发现程序中的问题并改进软件质量。设计良好的测试用例是保证测试有效性的关键。测试用例应该具有代表性和完备性，能够覆盖程序的各种情况和可能的错误情况。编写测试程序可以提高测试效率和准确性。自动化执行测试用例可以节省时间和精力，并能够快速验证程序的正确性。分析测试结果是发现问题和改进的关键步骤。通过仔细观察实际输出和预期输出之间的差异，我们可以定位问题并采取适当的措施。  完成这个黑盒测试的实验学习让我更加深入地理解了测试的重要性和技术。我尝试设计测试用例、编写测试程序，并通过实际测试发现和修复了程序中的问题。这对我的软件开发能力和质量意识的提升有一定帮助。 |