实验二 白盒测试用例设计实验

1. 引言

1.1 标识

本文档适用于以下测试环境

系统：win11，version 22H2

1.2 系统概述

本文档测试软件为“找零钱最佳组合程序”，具体功能如下：

1） 输入付款金额和商品价格，程序计算找零钱的最佳组合（找给顾客货币最少张数）

2） 商品价格应为一个整数且不大于 100 元。

3） 商品价格应为阿拉伯数字。

4）找零货币只有 50，20，10，5，2，1 六种面值。

2. 引用文件

本文档引用了《计算机软件文档编制规范》——GB/T 8567-2006

3. 测试执行结果

下面就以逻辑驱动测试中的判定/条件覆盖（CDC）和基本路径测试（BPC）两个方法为例进行白盒测试。

3.1 判定/条件覆盖（CDC）

3.1.1 基本原理

判定/覆盖测试是一种软件测试方法，它通过测试用例覆盖程序中的所有可能路径来确保程序的正确性。这种测试方法是基于逻辑回归分析的，它将程序视为一组逻辑表达式，并尝试组合这些表达式以覆盖所有可能的路径。为完成判定/覆盖测试，需要画出找零最佳组合函数流程图，如图 3-1 所示：

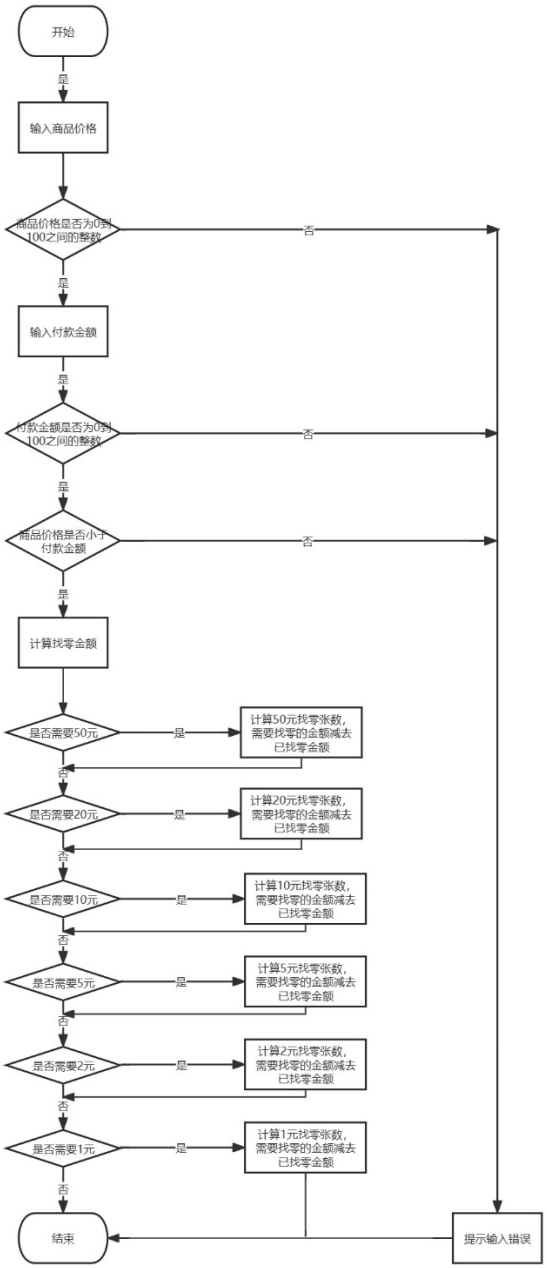


图 3-1 找零最佳组合函数流程图

3.1.2 采用“判定/条件覆盖”标准设计测试用例使得以下判定都取得两种结果

判定	条件	结果为真	结果为假
1	if (price < 0 || price > 100 || price % 1 != 0)	提示输入错误并结束	进入下一判定
2	if (payment < 0 || payment > 100 || payment % 1 != 0)	提示输入错误并结束	进入下一判定
3	if (payment < price)	提示输入错误并结束	进入下一判定
4	if ((payment - cost) / 50 > 0)	输出面值 50 的纸币张数	进入下一判定
5	if ((payment - cost) / 20 > 0)	输出面值 20 的纸币张数	进入下一判定
6	if ((payment - cost) / 10 > 0)	输出面值 10 的纸币张数	进入下一判定
7	if ((payment - cost) / 5 > 0)	输出面值 5 的纸币张数	进入下一判定
8	if ((payment - cost) / 2 > 0)	输出面值 2 的纸币张数	进入下一判定
9	if ((payment - cost) / 1 > 0)	输出面值 1 的纸币张数	结束

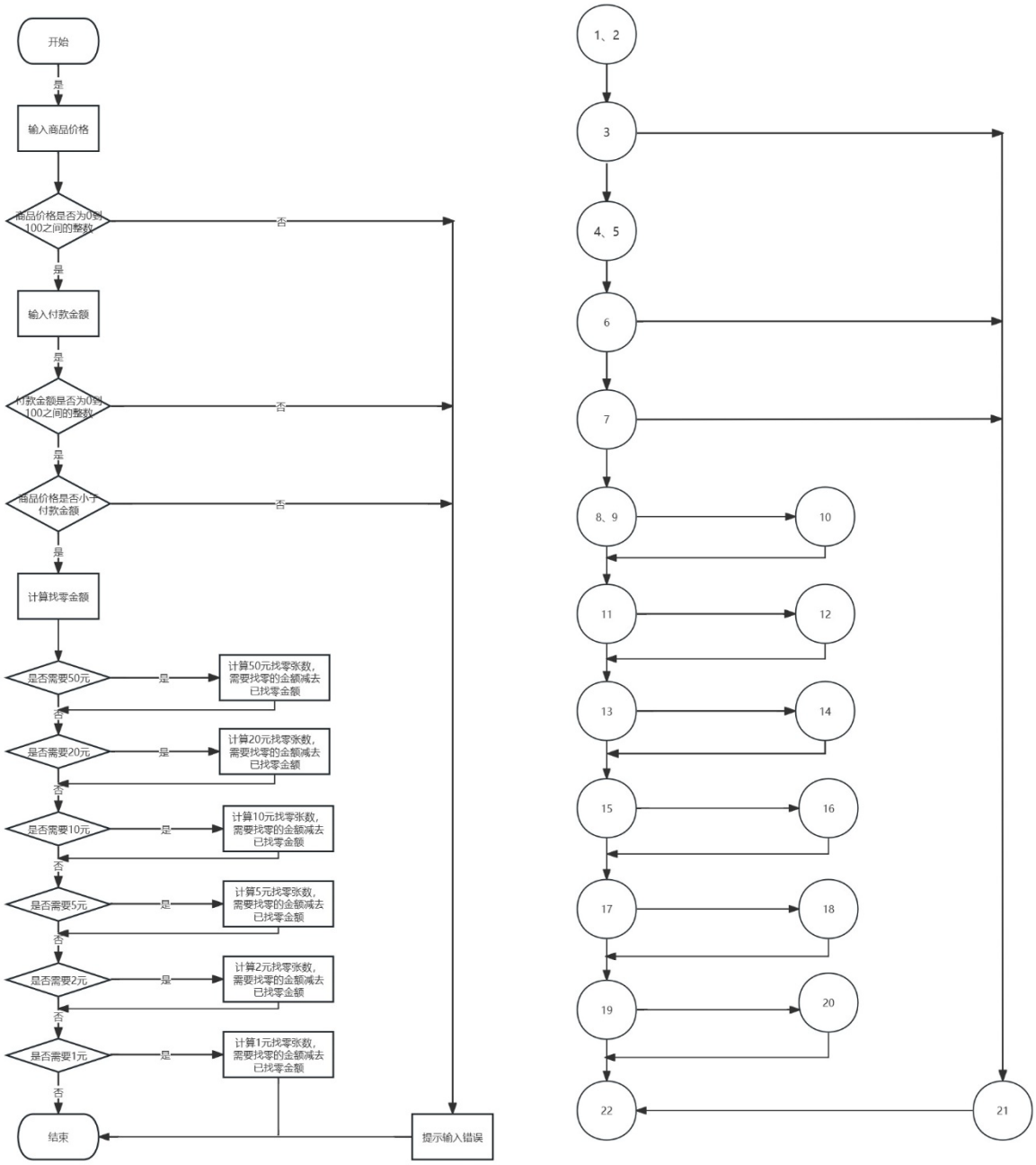
3.1.3 测试用例及执行结果

序号	输入	输出
1	付款金额:0.5	商品价格不符合要求!
2	付款金额:1002	商品价格不符合要求!
3	付款金额:10 商品价格:0.5	付款金额不符合要求!
4	付款金额:10 商品价格:-2	付款金额不符合要求!
5	付款金额:10 商品价格：20	付款金额必须大于或等于商品价格!
6	付款金额:100 商品价格:50	找零 50 元纸币 1 张
7	付款金额：100 商品价格：51	找零 20 元纸币 2 张 找零 5 元纸币 1 张 找零 2 元纸币 2 张
8	付款金额:50 商品价格:30	找零 20 元纸币 1 张

9	付款金额:50 商品价格:31	找零 10 元纸币 1 张 找零 5 元纸币 1 张 找零 2 元纸币 2 张
10	付款金额:50 商品价格:40	找零 10 元纸币 1 张
11	付款金额:50 商品价格:41	找零 5 元纸币 1 张 找零 2 元纸币 2 张
12	付款金额:100 商品价格:85	找零 10 元纸币 1 张 找零 5 元纸币 1 张
13	付款金额:100 商品价格:90	找零 10 元纸币
14	付款金额:50 商品价格:45	找零 5 元纸币 1 张
15	付款金额:50 商品价格:46	找零 2 元纸币 2 张
16	付款金额:100 商品价格:98	找零 2 元纸币 1 张
17	付款金额:100 商品价格:97	找零 2 元纸币 1 张 找零 1 元纸币 1 张

3.2 基本路径测试（BPC）

3.2.1 控制流图与圈复杂度



上述流程图中共有 9 个判定节点，所以圈复杂度为 10

3.2.2 测试用例设计及执行结果

基本路径测试在程序控制图的基础上，导出基本可执行路径集合，从而设计测试用例。设计出的测试用例要保证语句覆盖。

用例序号	输入	预期输出	覆盖路径	输出结果
1	商品价格为 0.5	输入错误	1-2-3-21-22	商品价格不符合要求!
2	付款金额为-10	输入错误	1-2-3-4-5-6-21-22	付款金额不符合要求!
3	商品价格为 20 付款金额为 10	输入错误	1-2-3-4-5-6-7-21-22	付款金额必须大于商品价格!
4	商品价格为 10 输入金额为 60	找零 50 元一张	1-2-3-4-5-6-7-8-9-10-11-13-15-17-19-22	找零 50 元纸币一张
5	商品价格为 40 输入金额为 60	找零 20 元一张	1-2-3-4-5-6-7-8-9-11-12-13-15-17-19-22	找零 20 元纸币一张
6	商品价格为 70 输入金额为 80	找零 10 元一张	1-2-3-4-5-6-7-8-9-11-13-14-15-17-19-22	找零 10 元纸币一张
7	商品价格为 55 输入金额为 60	找零 5 元一张	1-2-3-4-5-6-7-8-9-11-13-15-16-17-19-22	找零 5 元纸币一张
8	商品价格为 78 输入金额为 80	找零 2 元一张	1-2-3-4-5-6-7-8-9-11-13-15-17-18-19-22	找零 2 元纸币一张
9	商品价格为 59 输入金额为 60	找零 1 元一张	1-2-3-4-5-6-7-8-9-11-13-15-17-19-20-22	找零 1 元纸币一张
10	输入金额为 20 商品价格为 20	不用找零	1-2-3-4-5-6-7-8-9-11-13-15-17-19-22	无需找零！

4. 测试结果概述

4.1 对被测试软件的总体评估

软件应该能够正确处理用户的查询请求，对于非法的输入应该进行判断并进行错误提示。

4.2 测试环境的影响

与具体的系统环境无关。

4.3 改进建议

测试用例覆盖程度有待改进。

5. 软件评价

5.1 整体评价

基本实现计算找零钱最佳组合的功能，软件简洁且操作性较好，健壮性一般，可移植不足。

5.2 优化建议

建议添加更详细的输入错误提示，方便用户了解具体输入错误的原因。

6. 测试记录

测试项目	测试日期	测试时间	测试人员	测试环境
功能测试	2023.11.11	16:00	杨帅棋	华硕无畏 pro15 2022 win11，version 22H2