# 实验二 白盒测试用例设计实验

# 引言

## 标识

本文档适用于以下测试环境

系统：Windows 11

## 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。

# 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

# 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）

## 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：

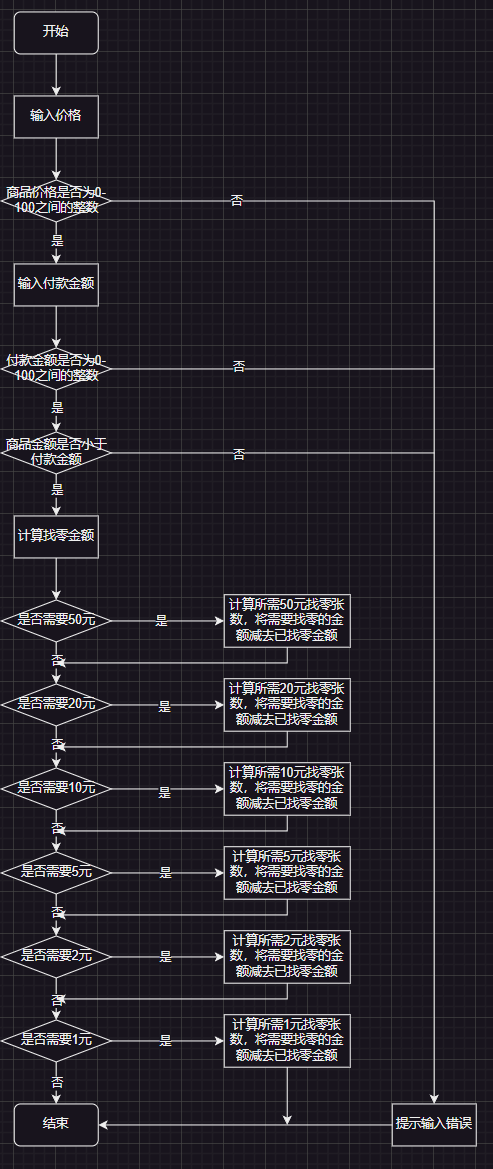


图3-1 找零最佳组合函数流程图

## 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 |
|  | if (price < 0 || price > 100 || price % 1 != 0) | 提示输入错误并结束 | 进入下一判定 |
|  | if (payment < 0 || payment > 100 || payment % 1 != 0) | 提示输入错误并结束 | 进入下一判定 |
|  | if (payment < price) | 提示输入错误并结束 | 进入下一判定 |
|  | if ((payment - cost) / 50 > 0) | 输出面值50的纸币张数 | 进入下一判定 |
|  | if ((payment - cost) / 20 > 0) | 输出面值20的纸币张数 | 进入下一判定 |
|  | if ((payment - cost) / 10 > 0) | 输出面值10的纸币张数 | 进入下一判定 |
|  | if ((payment - cost) / 5 > 0) | 输出面值5的纸币张数 | 进入下一判定 |
|  | if ((payment - cost) / 2 > 0) | 输出面值2的纸币张数 | 进入下一判定 |
|  | if ((payment - cost) / 1 > 0) | 输出面值1的纸币张数 | 结束 |

## 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）

1. 图片包含 游戏机, 链, 钟表

   描述已自动生成电脑萤幕画面

   低可信度描述已自动生成

## 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**

## 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！ |

# 测试结果概述

## 4.1 对被测试软件的总体评估

该程序能够正确处理用户的请求，对于非法的输入能够进行判断并进行错误提示

## 4.2 测试环境的影响

与系统环境无关。

## 4.3 改进建议

测试用例覆盖范围应该更广。

# 软件评价

## 5.1整体评价

程序基本满足用户需求，操作简单，提示完备，健壮性好，能够处理非法输入。

## 5.2优化建议

1. 使用不便，需要在IDE中才能运行
2. 付款金额的输入只设置了0-100的限制，没有考虑实际情况下不合理的输入，如：商品价格98元，实际情况下付款金额只能是98元或是100元，不可能是99元。
3. 对于输入范围的限制过于严格，上限100元过低。

# 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 | 2023.11.17 | 16：40 | 赵乐君 | Windows 11  AMD Ryzen 9 7940HX |