****

**软件测试技术第一次实验报告**



**学 院 智能与计算学部**

**专 业 软件工程**

**年 级 16**

**姓 名 李沛霖**

**2019年 3 月 12 日**

# 软件测试技术第一次实验报告

1. 需求分析（描述具体需求）

写一个java程序完成问题，并使用junit完成对代码的测试。问题要求如下：

你的口袋里有一张50,一张20和两张5元的纸币，以及三枚1元的硬币。编写一个程序来验证你能否给出规定的钱数（x）。

1. 概要设计（简单描述设计思路，配合UML图）

源程序：

第一步，按照从大到小的顺序录入自己拥有的所有金钱的种类；

第二步，检查异常情况，如满足超出总资产或者为负值时直接抛出否定结果；

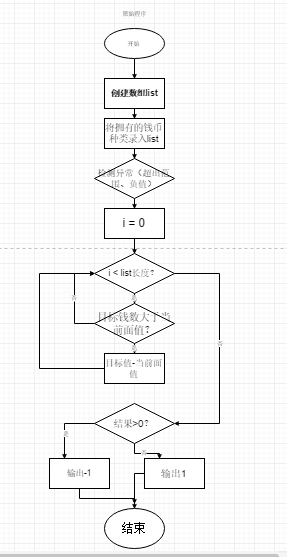
第三步，运用循环逐步将需要支付金钱的顺序排列出来；

第四步，如果可以实现目标结果，则返回1，否则返回0。

测试程序：

第一步，输入测试用例；

第二步，输入期望输出结果，得到输出。



1. 详细设计（详细描述具体如何实现，附代码及说明）

详情见代码注释。

源文件代码：

**package** cash;

**public** **class** Cash {

**int**[] list1 = {50,20,5,5,1,1,1};//从大到小输入现有的金钱种类

**public** **int** cash(**int** num) {

**int** remain = 0;//剩余钱数

**for**(**int** i = 0;i <list1.length;i++) {

remain += list1[i];

}//计算目前资产总额

**if**(num > remain) {

**return** -1;

}//目标比总额还多，返回异常

**else** **if**(num < 0){

**return** -1;

}//给出样例为负时，返回异常

**for**(**int** i = 0;i <list1.length;i++) {

**if**(num >= list1[i]) {

num -= list1[i];

}

}//循环测试，从大到小依次给出支付排列方式

**if**(num == 0) {

**return** 1;

}//若能成功给出目标数值，结果为0，返回1

**else** {

**return** -1;

}//若不能成功给出目标数值，结果不为0，返回-1

}

测试文件代码：

package cash;

import org.junit.Assert;

import org.junit.Before;

import org.junit.Test;

public class CashTest {

Cash c = null;

int[] num = {-10,0,3,9,20,49,52,53,54};//录入测试用例

int[] equal = {-1,1,1,-1,1,-1,1,1,-1};//录入期望结果

@Before

public void testBeforeClass(){

c = new Cash();//定义好测试前所需的类

}

@Test

public void testcash(){

for(int i = 0;i < num.length;i++) {

int result = c.cash(num[i]);

Assert.assertEquals(result,equal[i]);

}

//输入测试用例以及期望，进行测试

}

}

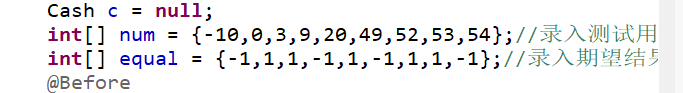
1. 调试分析（在实验过程中遇到的问题以及如何解决）

1在一开始编写测试程序时，我在Test中直接产生目标类，但是实际情况中这种做法会造成测试资源的浪费，所以将其单独列入before

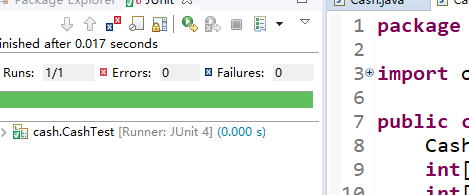
2 源程序中我一开始是直接录入所有金钱种类，在程序直接调取数值，但是这种方法的适用性明显仅限于这个问题，因此我在后来将金钱录入方式改为数组，并在之后调用数组中的元素。

1. 测试结果（描述输入和输出）

测试样例与期望（输入）：



结果：



1. 总结

在这次实验过程中，我学会了junit进行程序测试的基本方法，完成了该实验的基本任务，但是仍然有很多知识盲区和疑点，这些我会在之后的学习中逐渐补全。