习 题

主观题 10分

试用等价类划分法设计测试用例。

某城市的电话号码由三部分组成。这三部分的名称和内容分别是

地区码：空白或三位数字；

前　　缀：非’0’或’1’开头的三位数；

后　　缀：四位数字。

假定被测试的程序能接受一切符合上述规定的电话号码，拒绝所有不符合规定的号码，试用等价类划分法来设计它的测试用例。

输入条件	有效等价类	无效等价类
地区码	空白(1)，3 位数字(2)	有非数字字符(5)，少于 3 位数字(6)，多于 3 位数字(7)
前缀	从 200 到 999 之间的 3 位数字(3)	有非数字字符(8)，起始位为"0"(9)，起始位为"1"(10)，少于 3 位数字(11)，多于 3 位数字(12)
后缀	4 位数字(4)	有非数字字符(13)，少于 4 位数字(14)，多于 4 位数字(15)

主观题 10分

软件规格说明：某学校的学生公寓有 14 栋楼，用 A～N 这 14 个大写字母的其中一个代表楼号。每栋楼的层数为六层，代号为 1～6。每层楼有 40 个房间，编号为 01～40。具体表示一个宿舍房间时，用一个字母加三位数字表示，例如："C527"表示 C 楼第 5 层的 27 室。软件运行时，如果输入的房间号不在上述范围内，将不予接受，并显示输入无效。请根据规格说明，划分等价类。

有效等价类：

输入条件	有效等价类	无效等价类
宿舍号字符数	四位（1）	<4位（2），>4位（3）
楼号（首字符）	A～N（4）	O～Z（5），非大写字母字符（6）
层号（第2个字符）	1～6（7）	0（8），7～9（9），非数字字符（10）
房间编号（后两个字符）	01～40（11）	00（12），41～99（13），非数字字符（14）

主观题 10分

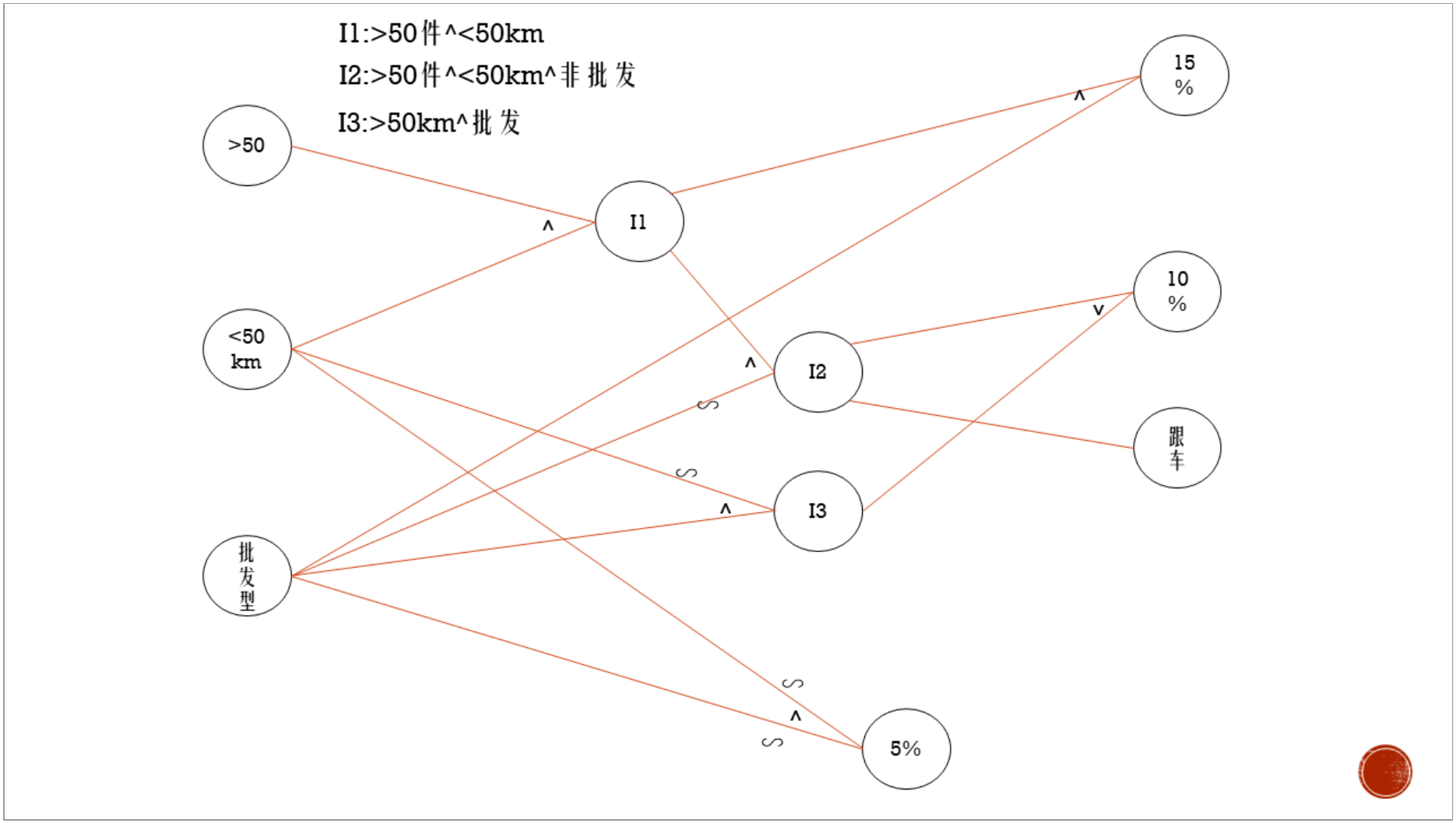
某单位财务管理系统中计算出差补助的方法是：
当员工办理长期出差时，不论是否出差，出差到哪里，每月固定补助1000；
当员工未办理长期出差时，如果出差省会城市，每月补助1500，非省会城市每月补助800，其他情况为0；
试用判定表法设计测试用例，测试系统的出差补助计算功能

	序号	1	2	3	4
条件桩	是否长期出差	1	0	0	0
	是否出差省会	-	1	0	0
	是否出差非省会	-	-	1	0
	其他	-	-	-	1
动作桩	补助1500		1		
	补助1000	1			
	补助500			1	
	补助0				1

主观题 10分

3.某销售系统的“供货折扣计算模块”，采用如下规则计算供货折扣：

当客户为批发型企业时，若订货数大于 50 件，发货距离不超过 50KM，则折扣率为 15%，而当发货距离超过 50KM，折扣率为 10%；当客户为非批发型企业时，若订货数大于 50 件，发货距离不超过 50KM，则折扣率为 10%，并派人跟车，而当发货距离超过 50KM 时，折扣率为 5%；画出因果图和判定表。


I1:>50件^<50km
I2:>50件^<50km^非批发
I3:>50km^批发
>50
<50 km
批发型
I1
I2
I3
15 %
10 %
跟车
5%
^
^
S
S
^
S
^
S
^
v

	序号				
原因	>50	1	1	1	1
	<50km	1	0	1	0
	批发型企业	1	1	0	0
	I1	1	0	1	0
	I2	0	0	1	0
	I3	0	1	0	0
结果	0.15	1			
	0.1		1	1	
	0.05				1
	跟车			1	

主观题 10分

```
public void dowork(int x,int y,int z){
  int k=0,j=0;
  if(x>3 && z<10){
    k=x*y-1;
    j=(int)Math.sqrt(k);
  }
  if (x==4||y>5){
       j=x*y+10;

  }
  j=j%3;

}
```

- 写出实现条件，判定，判定条件，条件组合覆盖的测试用例

主观题 10分

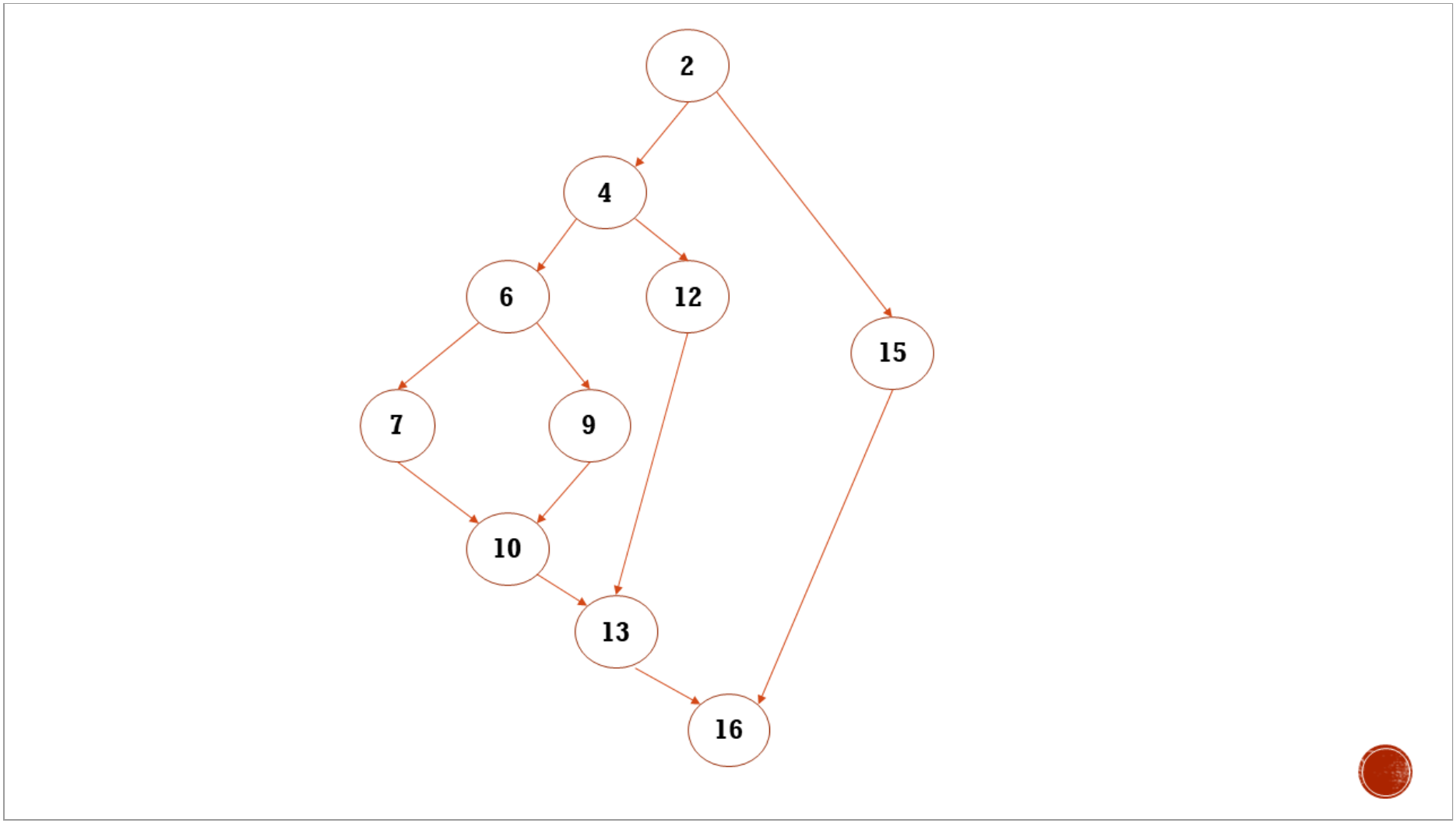
```
if(x>0 && y>0)
{
    magic = x+y+10; // 语句块1
}
else
{
    magic = x+y-10; // 语句块2
}

if(magic < 0)
{
    magic = 0;        // 语句块3
}
return magic;       // 语句块4
```

- 写出实现条件，判定，判定条件，条件组合覆盖的测试用例

主观题 10分

```
   Int IsLeap(int year)
1 {
2    if (year % 4 = = 0)
3     {
4       if (year % 100 == 0)
5       {
6           if ( year % 400 = = 0)
7                leap = 1;
8           else
9                leap = 0;
10       }
11     else
12          leap = 1;
13   }
14   else
15       leap = 0;
16   return leap;
17 }
```

（1）画出该程序的控制流图，并计算其圈复杂度。
（2）用基本路径覆盖法给出测试路径。
（3）为各测试路径设计测试用例


2
4
6
12
15
7
9
10
13
16

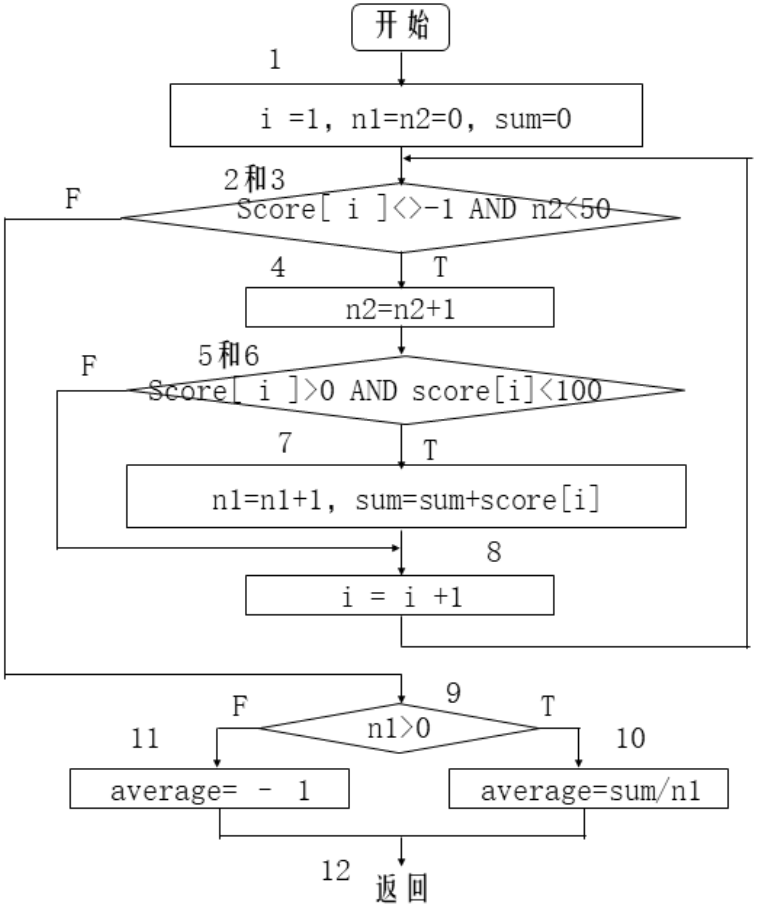
主观题 10分

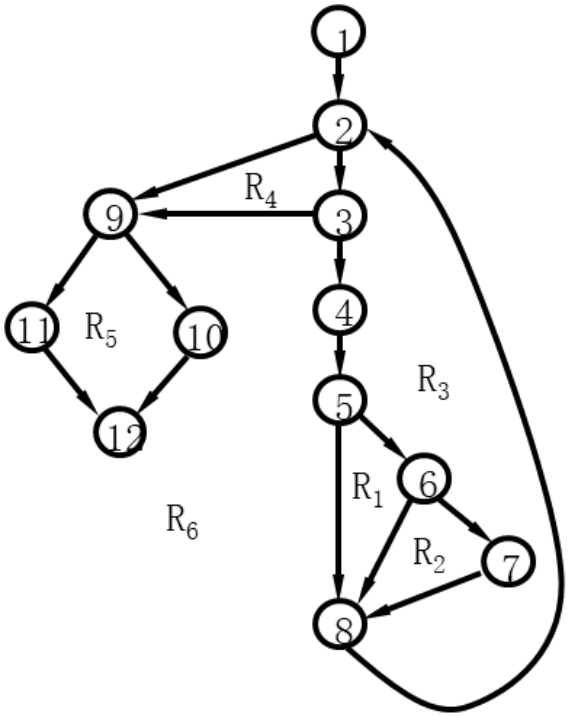
1画出流程图，并进行简化
2求解圈的度
3写出符合要求的路径集合

```
1   i=1,n1=n2=0,sum=0
                2                   3
    while (score[i]<>-1 and n2<50)
    {
4       n2=n2+1
                    5                       6
        if socre[i]>0 and score[i]<100
7               n1=n1+1，sum=sum+score[i]
8       i=i+1
    }
9   if n1>0
10          average=sum/n1
    else
11          average=-1
12  return
```

15

步骤1：导出过程的流图。


开始
1
i =1, n1=n2=0, sum=0
2和3
F
Score[i]<>-1 AND n2<50
4
T
n2=n2+1
5和6
F
Score[i]>0 AND score[i]<100
7
T
n1=n1+1, sum=sum+score[i]
8
i = i +1
9
F
n1>0
T
11
10
average= - 1
average=sum/n1
12
返回


1
2
3
4
5
6
7
8
9
10
11
12
R1
R2
R3
R4
R5
R6

步骤2:确定环形复杂性度量V(G):

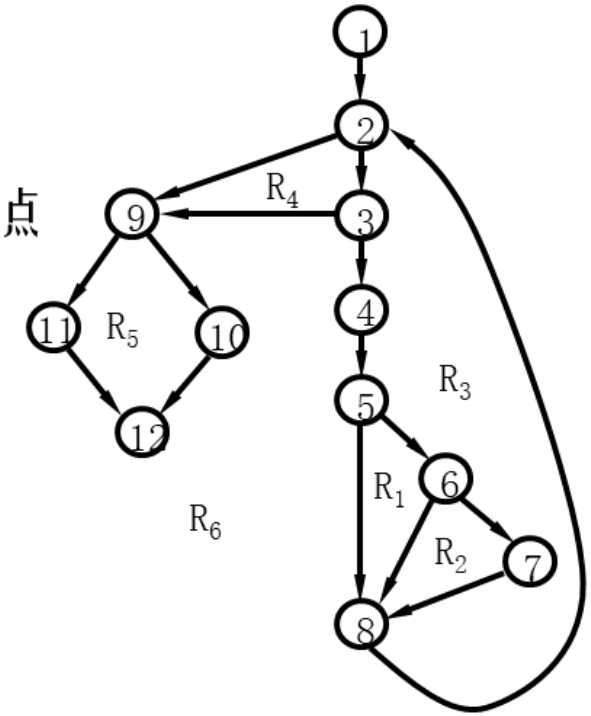
1）V(G)= 6 （个区域）

2）V(G)=E－N+2=16－12+2=6

其中E为流图中的边数，N为结点数；

3）V(G)=P+1=5+1=6

其中P为谓词结点的个数。在流图中，结点2、3、5、6、9是谓词结点。



步骤3：确定基本路径集合（即独立路径集合）。于是可确定6条独立的路径：

路径1：1-2-9-10-12

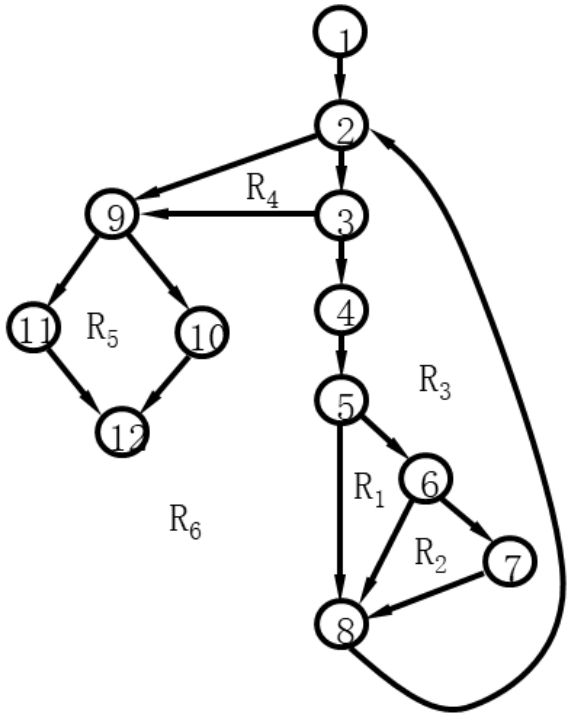
路径2：1-2-9-11-12

路径3：1-2-3-9-10-12

路径4：1-2-3-4-5-8-2…

路径5：1-2-3-4-5-6-8-2…

路径6：1-2-3-4-5-6-7-8-2…



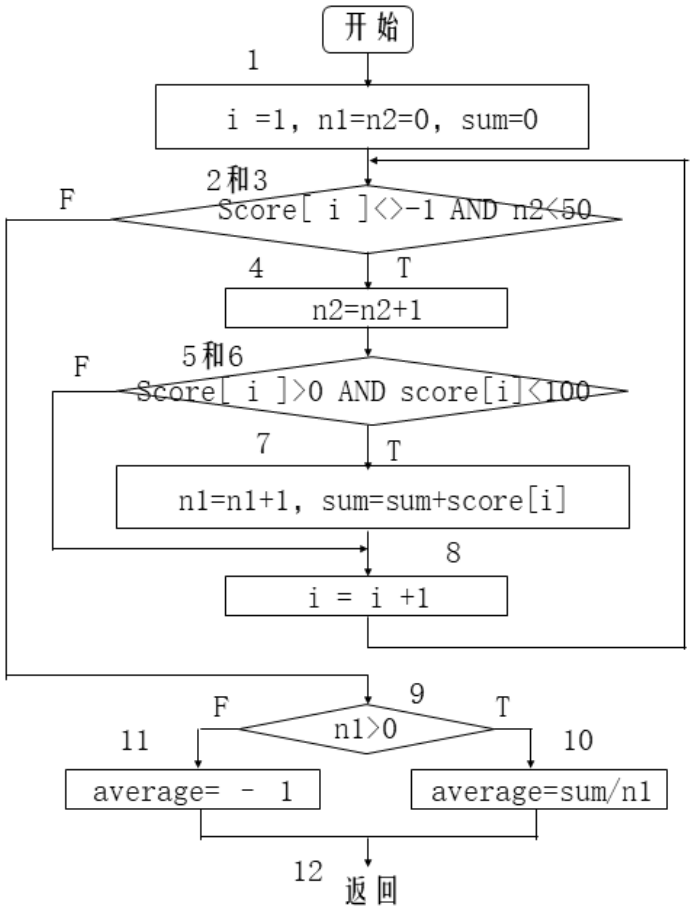
步骤4：为每一条独立路径各强设计一组测试用例，以便执行迫程序沿着该路径至少执行一次。

1）路径1(1-2-9-10-12)的测试用例：

score[k]=有效分数值，当k＜i；

score[i]=－1，2≤i≤50;

期望结果：根据输入的有效分数算出正确的分数个数n1、总分sum和平均分average。



2）路径2(1-2-9-11-12)的测试用例：

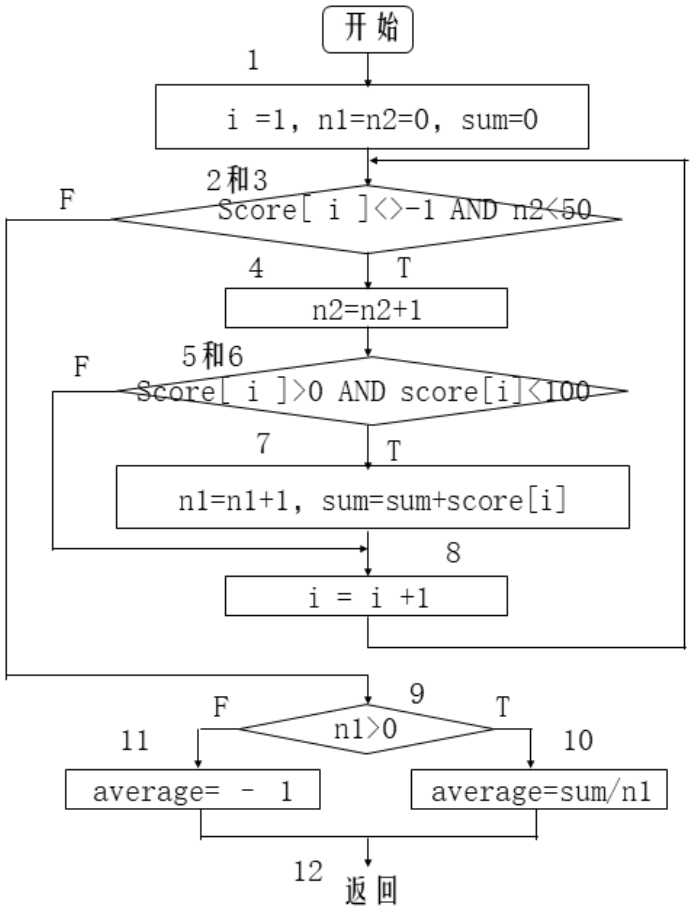
score[1]= - 1 ；

期望的结果：average = -1，其他量保持初值。

3）路径3(1-2-3-9-10-12)的测试用例：

输入多于50个有效分数，即试图处理51个分数，要求前51个为有效分数；

期望结果：n1=50、且算出正确的总分和平均分。



4）路径4(1-2-3-4-5-8-2…)的测试用例：

score[i]=有效分数，当i<50；

score[k]<0,k< i ；

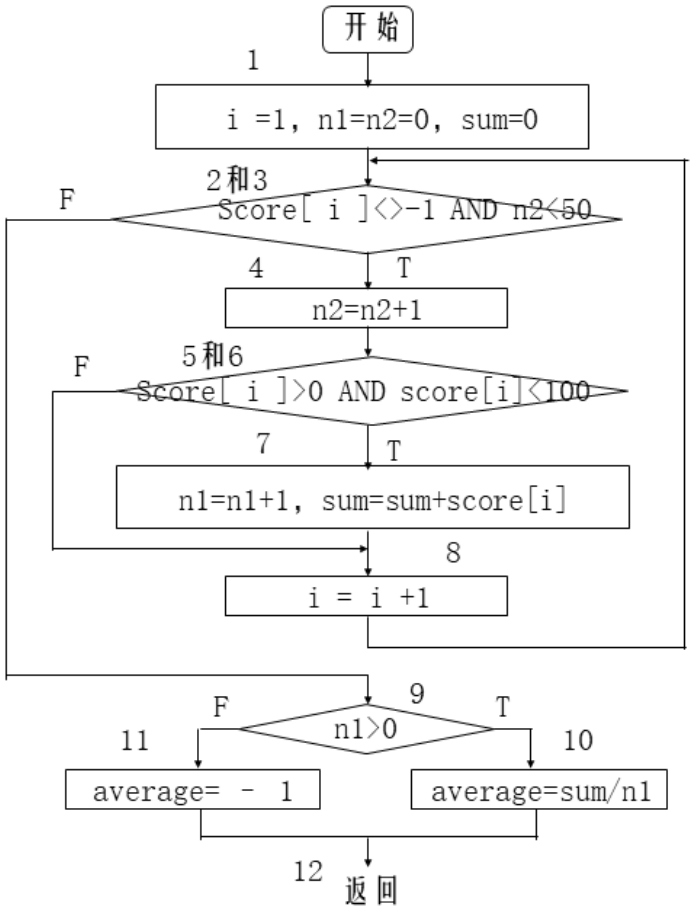
期望结果：根据输入的有效分数算出正确的分数个数n1、总分sum和平均分average。

5）路径5的测试用例：

score[i]=有效分数,当i<50;

score[k]>100,k< i ；

期望结果：根据输入的有效分数算出正确的分数个数n1、总分sum和平均分average。



6）路径6(1-2-3-4-5-6-7-8-2…)的测试用例：

score[i]=有效分数，当i<50;

期望结果：根据输入的有效分数算出正确的分数个数n1、总分sum和平均分average。