第5章 数 组

5.1 实验目的

(1) 理解并掌握数组的声明、创建、初始化和数组的遍历。
(2) 理解数组创建时的内存变化情况。
(3) 掌握经典的排序算法——冒泡排序和选择法排序。
(4) 掌握经典的二分查找的算法思路。
(5) 掌握 Arrays 类的使用。
(6) 掌握对象数组的创建和使用。
(7) 掌握二维数组的创建和使用。

5.2 实验任务

(1) 任务 1：成绩统计。
(2) 任务 2：食堂饭菜质量评价。
(3) 任务 3：打印杨辉三角形。

5.3 实验内容

5.3.1 任务 1 成绩统计

1. 任务目的

(1) 数组的声明、创建、初始化和数组的遍历。
(2) 能够灵活运用一维数组解决实际问题。

2. 任务描述

从键盘上输入若干学生(假设不超过 100)的成绩，计算平均成绩，并输出高于平均分的学生人数及成绩。这里约定输入成绩为 101 时结束。

3. 实施步骤

1) 算法分析

首先简要分析一下求平均分的算法思路。

step1：定义数组，数组的长度为 100。

step2：循环录入学生成绩并累加，如果录入成绩为 101，则跳出循环。

step3：平均分＝累加和/录入成绩的个数。

下面再简要分析一下输出高于平均分的学生人数及成绩的实现思路：设置一计数器，初始值为 0，遍历数组，发现高于平均分的就输出并对计数器加 1。

2）参考代码

```
public class Task1 {
    public static void main(String[] args) {
        float[] score=new float[100];
        Scanner input=new Scanner(System.in);
        float sum=0;                        //累加和
        int i;
        for(i=0;i<score.length;i++){
            System.out.print("请输入第"+(i+1)+"名的学生成绩：");
            score[i]=input.nextFloat();
            if(score[i]==101){
                break;
            }
            sum+=score[i];
        }
        float average=sum/i;
        System.out.println("平均分为："+average);
        int count=0;                        //统计高于平均分的学生人数
        for(int j=0;j<i;j++){
            if(score[j]>average){
                System.out.println("第"+(j+1)+"名的学生成绩为："+score[j]);
                count++;
            }
        }
        System.out.println("成绩高于平均分的有"+count+"人");
    }
}
```

3）运行程序

观察结果，如图 5-1 所示。

```
<terminated> Task1 (2) [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (2014-11-24 下午12:47:13)
请输入第1名的学生成绩：78
请输入第2名的学生成绩：67
请输入第3名的学生成绩：89
请输入第4名的学生成绩：99
请输入第5名的学生成绩：67
请输入第6名的学生成绩：56
请输入第7名的学生成绩：101
平均分为：76.0
第1名的学生成绩为：78.0
第3名的学生成绩为：89.0
第4名的学生成绩为：99.0
成绩高于平均分的有3人
```

图 5-1　成绩统计

4. 任务拓展

（1）输出最高分和最低分。

（2）成绩排序输出。

5.3.2 任务 2 食堂饭菜质量评价

1. 任务目的

能够灵活运用数组解决实际问题。

2. 任务描述

要求 20 名同学对学生食堂饭菜的质量进行 1～5 的评价(1 表示很差,5 表示很好)。将这 20 个结果输入整型数组,并对打分结果进行分析。

3. 实施步骤

1) 算法分析

我们希望统计出每个分数对应的学生人数。学生最终的打分情况可以借助于一个整型数组 answers 来保存。首先定义一个包含 20 个打分结果的 answers 数组,然后再定义一个包含 6 个元素的数组 frequency 来统计各种评价的次数,frequency 中的每个元素此时都被看成了一个得分的计数器,其默认的初始值为 0。在这里为何要将数组长度定义为 6 呢?我们想让 frequency[1]统计的是分值 1 的次数,frequency[2]统计的是分值 2 的次数,…,frequency[5]统计的是分值 5 的次数,这样正好一一对应起来,这里忽略掉 frequency[0]。比如读入第一个学生的评价,他的评价是 5 分,就将 frequency[5]的计数值加 1。当遍历完 answers 数组后,对应的 frequency 数组里面也已经统计完了。

2) 参考代码

```
public class Task2 {
    public static void main(String[] args) {
        int[] answers={3,1,2,5,4,2,2,3,4,5,1,2,3,4,2,1,3,2,
                4,2};
        int[] frequency=new int[6];
        for (int i=0; i<20; i++) {
            frequency[answers[i]]++;
        }
        System.out.println("分值\t 学生数");
        for (int i=1; i<6; i++) {
            System.out.println(i+"\t"+frequency[i]);
        }
    }
}
```

3) 运行程序

观察结果,如图 5-2 所示。

```
<terminated> Task2 (2) [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (2014-11-24 下午2:00:48)
分值    学生数
1       3
2       7
3       4
4       4
5       2
```

图 5-2 食堂饭菜质量评价

4. 任务拓展

输出对食堂饭菜质量的最终评价分数。

5.3.3 任务 3 打印杨辉三角形

1. 任务目的

(1) 理解并掌握二维数组的创建和使用。

(2) 能够灵活运用二维数组解决实际问题。

2. 任务描述

杨辉三角形(又称为贾宪三角形,帕斯卡三角形)是二项式系数在三角形中的一种几何排列。要求打印如图 5-3 所示的杨辉三角形。

```
<terminated> Task3 (2) [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (2014-11-24 下午2:21:55)
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
```

图 5-3 杨辉三角形

3. 实施步骤

1) 算法分析

将图 5-3 可以看成由行和列组成的,用 i 表示行,用 j 表示列,均从 0 开始,分析图 5-3 可以发现如下规律:

如果 j 为 0 时或者对角线上,即 i==j 时数字为 1。

其他情况 a[i][j]=a[i-1][j-1]+a[i-1][j]。

2) 参考代码

```
public class Task3 {
    public static final int ROW=6;                          //设置行数
    public static void main(String[] args) {
        int a[][]=new int[ROW][];
        for (int i=0; i<ROW; i++) {                         //循环初始化数组
            a[i]=new int[i+1];
        }
        for (int i=0; i<ROW; i++) {                         //循环行数
            for (int j=0; j<=a[i].length-1; j++) {          //在行基础上循环列数
                if (j==0 || i==j)
                    a[i][j]=1;                              //将两侧元素设为 1
                else
                    //元素值为其正上方元素与左上角元素之和
                    a[i][j]=a[i-1][j-1]+a[i-1][j];
            }
        }
        for (int i=0; i<ROW; i++) {                         //循环行数
            for (int j=0; j<=a[i].length-1; j++)
```

```
                //在行基础上循环列数
                System.out.print(a[i][j]+" ");          //输出
            System.out.println();                       //换行
        }
    }
}
```

3）运行程序

观察结果，如图 5-3 所示。

4. 任务拓展

输出等腰三角形形状的杨辉三角形。