内蒙古自治区“十四五”职业教育规划教材
校企双元合作开发“互联网 + 教育”新形态一体化系列教材

Java 程序设计案例教程

主　编　田　智　冯建平
副主编　荣艳冬　霍婕婷　冯子龙

合肥工業大學出版社
HEFEI UNIVERSITY OF TECHNOLOGY PRESS

图书在版编目（CIP）数据

Java程序设计案例教程 / 田智，冯建平主编.—合肥：合肥工业大学出版社，2023.2（2025.4重印）

ISBN 978-7-5650-6216-2

Ⅰ. ①J… Ⅱ. ①田… ②冯… Ⅲ. ①JAVA语言—程序设计—教材 Ⅳ. ①TP312.8

中国国家版本馆CIP数据核字（2023）第033952号

Java程序设计案例教程

JAVA CHENGXU SHEJI ANLI JIAOCHENG

田 智 冯建平 主编

责任编辑 张 慧
出版发行 合肥工业大学出版社
地 址 （230009）合肥市屯溪路193号
网 址 www.hfutpress.com.cn
电 话 人文社科出版中心：0551-62903205
营销与储运管理中心：0551-62903198
规 格 787毫米×1092毫米 1/16
印 张 23.5
字 数 540千字
版 次 2023年2月第1版
印 次 2025年4月第3次印刷
印 刷 三河市海新印务有限公司
书 号 ISBN 978-7-5650-6216-2
定 价 68.00元

前言

党的二十大提出“科技是第一生产力、人才是第一资源、创新是第一动力”。作为当下的计算机相关专业学生，大家需要努力学习技能、探索Java世界，提高自身素质和竞争力，以技能成才、技能报国为共同目标。

Java是一种面向对象的计算机编程语言，具有卓越的通用性、高效性、平台移植性和安全性。Java旨在让应用程序开发人员“一次编写，到处运行”，这意味着编译的Java代码可以在支持Java的所有平台上运行，而无须重新编译。

自从1995年首次发布以来，Java迅速崛起成为网络编程领域的先锋技术。后续版本的不断推出，进一步巩固了Java在这一领域的领导地位。至今，Java仍然是开发基于Web的应用程序的首选语言。更值得一提的是，Java在智能手机革命中也扮演了关键角色，特别是在Android编程领域的广泛应用。在全球云计算和大数据技术迅速发展的当下，Java展现出了显著的优势和广阔的应用前景。它不仅汇聚了强大的开发者专业社群，而且在无数开发者的心中占据了举足轻重的地位。随着技术的演进和市场的变化，Java的影响力和重要性持续攀升，成为当代技术生态中不可或缺的一环。

本书以Java8为基础，内容包括：认识Java、简单的Java程序、Java基本程序设计、数组与方法、类的基本形式、类的继承、异常处理、包及访问权限、多线程、文件（IO）操作、Java常用类库、Java网络程序设计、图形用户界面设计等。本书注重理论与实践相结合，内容详尽，并提供了大量实例，以突出应用能力的培养。在第10章、第12章、第13章中，我们精选了企业的案例，并将其抽象成学生容易上手的案例，从而使本书具有突出的实用性。书中将Java的语法知识融入简单明了的程序中，每一个小节后都有精选案例。在对程序进行解析时既给出正确的代码，也展示常见错误代码，方便分析出错原因，提高程序纠错能力，提高学习的效果。即使是没有编程基础的初学者，通过本书的学习，亦能达到一般编程开发人员的水平。

本书在所有章节中都设置了实训环节，将这些实训的内容有机地串联在一起，帮助学生在程序设计过程中学会触类旁通，从而提高开发效率。本书通过案例和实训中的“思政元素”，将党的二十大精神“立德树人”和“社会主义核心价值观”潜移默化地融入教学中，使学生不仅在

技能上具备广度和深度，更在素质上得到提升，以促进学生全面发展。

本书由田智、冯建平任主编，荣艳冬、霍婕婷、冯子龙任副主编。本书第3、5、6、10、13章由田智编写；第1、2章由冯建平编写；第4、7章由荣艳冬编写；第8、9章由霍婕婷编写；第11、12章由冯子龙编写；实训内容由金名信息技术股份有限公司的资深工程师指导编写。在本书付梓之际，对海川、高丽、邢苗苗、武书琴老师表示深深的谢意，案例的精选、习题的策划，诸位老师都给出了非常宝贵的意见和建议！

由于时间仓促，编者水平有限，书中难免有疏漏与不足之处，欢迎读者批评指正。

编　者

目录

第 4 章

数组与方法

若想要存放一连串相关的数据，使用数组是一个相当好用的选择。此外，如果某个程序片段反复出现，那么将它定义成一个方法可以有效地简化程序代码。本章的中心内容是介绍数组的基本用法与方法的应用，对数组与方法的使用有更深一层的认识。

数组是由一组相同类型的变量所组成的数据类型，它们以一个共同的名称表示，数组中的个别元素则以标注来表示其存放的位置。数组依照存放元素的复杂程度分为一维数组、二维数组和多维数组。本章先从一维数组谈起。

4.1 一维数组

一维数组可以存放上千万个数据，并且这些数据的类型是完全相同的。

4.1.1 一维数组的声明与内存的分配

要使用 Java 的数组，必须经过两个步骤：一是声明数组，二是分配内存给该数组。这两个步骤的语法如下。

格式 4-1 一维数组的声明与分配内存

```
数据类型     数组名[];            //声明一维数组
数组名 =  new 数据类型[个数];  //分配内存给数组
```

数组的声明格式里，“数据类型”是声明数组元素的数据类型，常见的类型有整型、浮点型与字符型等。“数组名”是用来统一这组相同数据类型的元素的名称，其命名规则和变量相同，应使用有意义的名称为数组命名。数组声明后，接下来便是要配置数组所需的内存，其中“个数”是告诉编译器，所声明的数组要存放多少个元素，而“new”则是命令编译器根据括号里的个数，在内存中开辟一块内存供该数组使用。下面是关于一维数组的声明并分配内存给该数组的一个范例：

```
1.  int  score[];          //声明整型数组 score
2.  score = new int[3];    //为整型数组 score 分配内存空间，其元素个数为 3
```

在上例中的第 1 行，当声明一个整型数组 score 时，score 可视为数组类型的变量，此时这个变量并没有包含任何内容，编译器仅会分配一块内存给它，用来保存指向数组实体的地址，如图 4-1 所示。

声明之后，接着要做内存分配的操作，也就是上例中第2行语句。这一行会开辟3 个可供保存整数的内存空间，并把此内存空间的参考地址赋给 score 变量。其内存分配的流程如图4-2 所示。

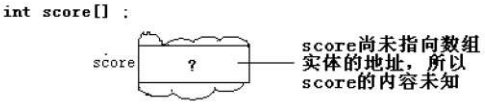

图4-1　声明整型数组

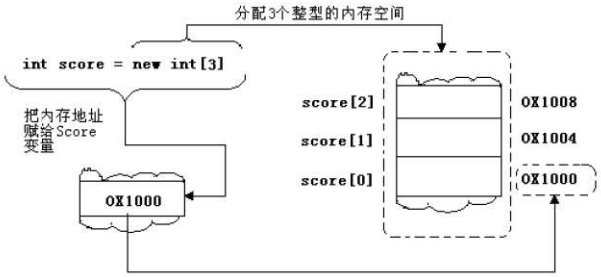

图4-2　内存分配

图4-2中的内存参考地址 OX1000 是假赋值，此值会因环境的不同而异。如第 3 章中所述，数组是属于非基本数据类型，因此数组变量 score 所保存的并非是数组的实体，而是数组实体的参考地址。除了用格式 4-1 声明并分配内存给数组之外，也可以用较为简洁的方式，把两行缩成一行来编写，其格式如下。

格式4-2　声明数组的同时分配内存

```
数据类型 数组名[] = new 数据类型[个数]
```

上述的格式会在声明的同时，即分配一块内存空间，供该数组使用。下面的范例是声明整型数组 score，并开辟可以保存 11 个整数的内存给 score 变量。

```
1.  int score[] = new int[11];
2.  //声明一个元素个数为 11的整型数组 score，同时开辟一块内存空间供其使用。
```

在Java中，由于整数数据类型所占用的空间为4个bytes，而整型数组score可保存的元素有11个，所以上例中占用的内存共有4×11=44个字节。图4-3是将数组score用图形来表示，比较容易理解数组的保存方式。

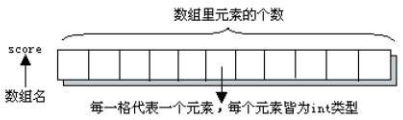

图4-3　数组的保存方式

4.1.2 数组中元素的表示方法

想要使用数组里的元素，可以利用索引来完成。Java 的数组索引编号由 0 开始，以4.1.1节中的 score 数组为例，score[0]代表第 1 个元素，score[1]代表第 2 个元素，score[10]为数组中第 11 个元素（也就是最后一个元素）。接下来，看一个范例。下面的程序里，声明了一个一维数组，其长度为 3，利用for 循环输出数组的内容后，再输出数组的元素个数。

范例　TestJava4_1.java

```
1.  //下面这段程序说明了一维数组的使用方法
2.      public class TestJava4_1
3.      {
4.          public static void main(String args[])
5.          {
6.              int i;
7.              int a[];          //声明一个整型数组 a
```

```
8.              a=new int[3];    //开辟内存空间供整型数组a使用，其元素个数为3
9.              for(i=0;i<3;i++)//输出数组的内容
10.                 System.out.print("a["+i+"] = "+a[i]+",\t");
11.             System.out.println("\n 数组长度是: "+a.length);
                                    //输出数组长度
12.         }
13.     }
```

输出结果如图4-4所示。

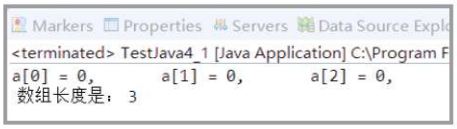

图4-4 TestJava4_1.java的运行结果

● 程序说明

（1）第 7 行声明整型数组 a；第 8 行开辟了一块内存空间，以供整型数组 a 使用，其元素个数为 3。

（2）第 9~10 行，利用 for 循环输出数组的内容。由于程序中并未给予数组元素赋值，因此输出的结果都是 0。

（3）第 11 行输出数组的长度。此例中数组的长度是 3，即代表数组元素的个数有 3 个。

要特别注意的是，在 Java 中取得数组的长度（也就是数组元素的个数）可以利用“.length”完成，如下面的格式。

格式4-3 数组长度的取得

```
数组名.length
```

也就是说，若要取得 TestJava4_1.Java 中所声明的数组 a 的元素个数，只要在数组a的名称后面加上“.length”方法即可，如下面的程序片段：

```
a.length;        //取得数组的长度
```

4.1.3 数组初值的赋值

如果想直接在声明时就给数组赋初值，可以利用大括号完成。只要在数组的声明格式后面再加上初值的赋值即可，如下面的格式。

格式4-4 数组赋初值

```
数据类型 数组名[] = {初值 0，初值 1，…，初值 n}
```

在大括号内的初值会依序指定给数组的第 1，…，n+1 个元素。此外，在声明的时候，并不需要将数组元素的个数列出，编译器根据所给出的初值个数来判断数组的长度。如下面的数组声明及赋初值范例：

```
int day[] = {32,23,45,22,13,45,78,96,43,32};   //数组声明并赋初值
```

在上面的语句中，声明了一个整型数组 day，虽然没有特别指明数组的长度，但是由于大括号里的初值有 10 个，编译器会分别依序指定给各元素存放，day[0] 为 32，day[1] 为 23，…，day[9] 为 32。

范例 TestJava4_2.java

```
1.  //一维数组的赋值，这里采用静态方式赋值
2.      public class TestJava4_2
```

```
3.      {
4.          public static void main(String args[])
5.          {
6.              int i;
7.              int a[]={5,6,8};            //声明一个整数数组 a 并赋初值
8.              for(i=0;i<a.length;i++)     //输出数组的内容
9.                  System.out.print("a["+i+"]="+a[i]+",\t");
10.             System.out.println("\n 数组长度是:"+a.length);
11.         }
12.     }
```

输出结果如图4-5所示。

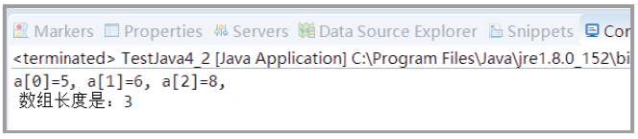

图4-5 TestJava4_2.java的运行结果

除了在声明时就赋初值之外，也可以在程序中为某个特定的数组元素赋值。可以将程序 TestJava4_2.java 的第7行更改成下面的程序片段：

```
1. int a [] = new int[];
2. a[0] = 5;
3. a[1] = 6;
4. a[2] = 8;
```

4.1.4 与数组操作有关的 API 方法

Java 语言中提供了许多的 API 方法，供开发人员使用。下面介绍两种常用的数组操作方法，一个是数组的复制操作，另一个是数组的排序操作。其他的操作可查阅 JDK 帮助文档。

范例 TestJava4_3.java

```
1. //以下这段程序说明数组的复制操作
2.     public class TestJava4_3
3.     {
4.     public static void main(String[] args)
5.     {
6.         int a1[] = {1,2,3,4,5}; //声明两个整型数组 a1、a2，并进行静态初
                                   //始化
7.         int a2[] = {9,8,7,6,5,4,3};
8.         System.arraycopy(a1,0,a2,0,3);       //执行数组复制的操作
9.         System.out.print("a1 数组中的内容:");
10.        for(int i=0;i<a1.length;i++)         //输出 a1 数组中的内容
11.            System.out.print(a1[i]+" ");
12.        System.out.println();
13.        System.out.print("a2 数组中的内容:");
14.        for(int i=0;i<a2.length;i++)         //输出 a2 数组中的内容
15.            System.out.print(a2[i] +" ");
16.        System.out.println("\n 数组拷贝完成! ");
17.    }
18. }
```

输出结果如图4-6所示。

图4-6 TestJava4_3.java的运行结果

● 程序说明

System.arrayCopy(source,0,dest,0,x)：语句的意思是，复制源数组从下标 0 开始的 x 个元素到目标数组，从目标数组的下标 0 所对应的位置开始存取。

范例 TestJava4_4.java

```
1.  //以下程序是数组的排序操作，在这里使用了 sort 方法对数组进行排序
2.      import java.util.*;
3.      public class TestJava4_4
4.      {
5.          public static void main(String[] args)
6.          {
7.              int a[] = {4,32,45,32,65,32,2};
8.              System.out.print("数组排序前的顺序：");
9.              for(int i=0;i<a.length;i++)
10.                 System.out.print(a[i]+" ");
11.             Arrays.sort(a);         //数组的排序方法
12.             System.out.print("\n 数组排序后的顺序：");
13.             for(int i=0;i<a.length;i++)
14.                 System.out.print(a[i]+" ");
15.         }
16.     }
```

输出结果如图4-7所示。

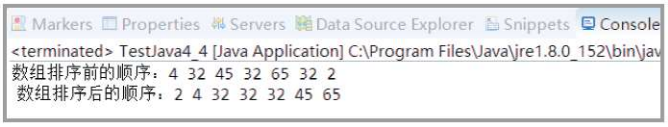

图4-7 TestJava4_4.java的运行结果

● 程序说明

程序第 11 行的 Arrays.sort(数组名)为数组排序的操作，但这个方法在 java.util 包里面，所以在用到的时候需要先将它导入，至于包的概念，本书在以后的章节中会讲到。

案例1 找出数组元素中的最大值与最小值

【案例描述】

求得数组中的最大值和最小值。

【技术要点】

数组的索引就好像饭店房间的编号一样，想要找到某个房间时，就得先找到房间编号。

再举一个范例，说明如何将数组里的最大值和最小值列出。

【代码与注释】

范例 MaxMin.java

```
1.  //这个程序主要是求得数组中的最大值和最小值
2.      public class MaxMin
3.      {
4.          public static void main(String args[])
5.          {
6.              int i,min,max;
7.              int A[]={74,48,30,17,62}; //声明整数数组 A,并赋初值
8.              min=max=A[0];
9.              System.out.print("数组 A 的元素包括: ");
10.             for(i=0;i<A.length;i++)
11.             {
12.                 System.out.print(A[i]+" ");
13.                 if(A[i]>max)            //判断最大值
14.                     max=A[i];
15.                 if(A[i]<min)            //判断最小值
16.                     min=A[i];
17.             }
18.             System.out.println("\n 数组的最大值是:"+max);
                                            //输出最大值
19.             System.out.println("数组的最小值是:"+min);
                                            //输出最小值
20.         }
21.     }
```

【运行结果】

输出结果如图4-8所示。

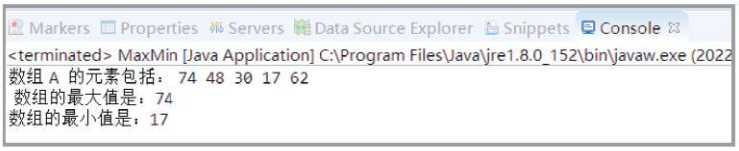

图4-8 MaxMin.java的运行结果

【相关知识】

(1)第 6 行声明整数变量 i 作为循环控制变量及数组的索引;声明存放最小值的变量 min 与最大值的变量 max。

(2)第 7 行声明整型数组 A,其数组元素有 5 个,其值分别为 74、48、30、17、62。

(3)第 8 行将 min 与 max 的初值设为数组的第一个元素。

(4)第 10~17 行逐一输出数组里的内容,并判断数组里的最大值与最小值。

(5)第 18~19 行输出比较后的最大值与最小值。

将变量 min 与 max 初值设成数组的第一个元素后,再逐一与数组中的各元素相比。比 min 小,就将该元素的值指定给 min 存放,使 min 的内容保持最小;同样的,当该元素比 max 大时,就将该元素的值指定给 max 存放,使 max 的内容保持最大。for 循环执行完,表示数组中所有的元素都已经比较完毕,此时变量 min 与 max 的内容就是最小值与最大值。

4.2 二维数组

虽然一维数组可以处理一般简单的数据，但是在实际的应用中仍显不足，所以 Java 也提供了二维数组及多维数组。

4.2.1 二维数组的声明与分配内存

二维数组声明的方式和一维数组类似，内存的分配也一样是用 new 关键字。其声明与分配内存的格式如下。

格式 4-5 二维数组的声明格式

```
数据类型      数组名[][];
数组名  = new   数组类型[行的个数][列的个数];
```

与一维数组不同的是，二维数组在分配内存时，必须告诉编译器二维数组行与列的个数。因此在格式 4-5 中，“行的个数”是告诉编译器所声明的数组有多少行，“列的个数”则是说明该数组有多少列，如下面的范例：

```
1.  int score[][];            //声明整型数组 score
2.  score = new int[4][3];    //配置一块内存空间，供 4 行 3 列的整型数组
score 使用
```

同样的，可以用较为简洁的方式来声明数组，其格式如下。

格式 4-6 二维数组的声明格式

```
数据类型 数组名[][] = new 数据类型[行的个数][列的个数] ;
```

若用上述的写法，则是在声明的同时，开辟一块内存空间，以供该数组使用，如下：

```
int score[][] = new int[4][3]; //声明整型数组score，同时为其开辟一块内存空间
```

上面的语句中，整型数据 score 可保存的元素有 4×3 = 12 个，而在 Java 中，int 数据类型所占用的空间为 4 个字节。因此该整型数组占用的内存共为 4×12 = 48 个字节。

如果想直接在声明时就为数组赋初值，可以利用大括号完成。只需在数组的声明格式后面再加上所赋初值，如下面的格式。

格式 4-7 二维数组赋初值的格式

```
数据类型     数组名[][]  =  {
                                {第0行初值},
                                {第1行初值},
                                …
                                {第n行初值}
                                }
```

要特别注意的是，用户不需要定义数组的长度。因此在数组名后面的中括号里不必填入任何的内容。此外，在大括号内还有几组大括号，每组大括号内的初值会依序指定给数组的第 0，1，…，

n 行元素。如下面的关于数组 num 声明及赋初值的范例：

```
1.  int num[][] = {
2.  {23,45,21,45},              //二维数组的初值赋值
3.  {45,23,46,23}
4.  };
```

上面的语句声明了一个整型数组 num，数组有 2 行 4 列共 8 个元素，大括号里的几组初值会分别依序指定给各行里的元素存放，num[0][0] 为 23，num[0][1] 为45，…，num[1][3] 为 23。

1. 每行的元素个数不同的二维数组

Java 允许二维数组中每行的元素个数均不相同，这点与一般的程序语言是不同的。例如，下面的语句是声明整型数组 num 并赋初值，而初值的赋值指明了 num 具有三行元素，其中第一行有 4 个元素，第二行有 3 个元素，第三行则有 5 个元素：

```
1.  int num[][] = {
2.  {42,54,34,67},
3.  {33,34,56},
4.  {12,34,56,78,90}
5.  };
```

2. 取得二维数组的行数与特定行的元素的个数

在二维数组中，若是想取得整个数组的行数，或者是某行元素的个数时，可利用“.length”来获取，其语法格式如下。

格式4-8 取得二维数组的行数与特定行的元素的个数

```
数组名.length                //取得数组的行数
数组名[行的索引].length      //取得特定行元素的个数
```

也就是说，如要取得二维数组的行数，用数组名加上“.length”即可；如要取得数组中特定行的元素的个数，则须在数组名后面加上该行的索引值，再加上“.length”，如下面的程序片段：

```
1.  num.length;              //计算数组 num 的行数，其值为 3
2.  num[0].length            //计算数组 num 的第 1 行元素的个数，其值为 4
3.  num[2].length            //计算数组 num 的第 3 行元素的个数，其值为 5
```

4.2.2 二维数组元素的引用及访问

二维数组元素的输入与输出方式与一维数组相同，看下面这个范例。

范例 TestJava4_5.java

```
1.  //二维数组的使用说明，这里采用静态赋值的方式
2.   public class TestJava4_5
3.      {
4.      public static void main(String args[])
5.      {
6.          int i,j,sum=0;
7.          int num[][]={{30,35,26,32},{33,34,30,29}};
```

```
                                                    //声明数组并设置初值
8.          for(i=0;i<num.length;i++)   //输出销售量并计算总销售量
9.          {
10.             System.out.print("第 "+(i+1)+" 个人的成绩为:");
11.             for(j=0;j<num[i].length;j++)
12.             {
13.                 System.out.print(num[i][j]+" ");
14.                 sum+=num[i][j];
15.             }
16.             System.out.println();
17.         }
18.     System.out.println("\n 总成绩是 "+sum+" 分! ");
19.          }
20.  }
```

输出结果如图4-9所示。

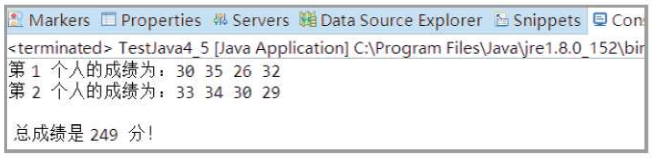

图4-9 TestJava4_5.java的运行结果

● 程序说明

(1)第6行声明整数变量i、j作为外层与内层循环控制变量及数组的索引，i控制行的元素，j控制列的元素；而sum则用来存放所有数组元素值的和，也就是总成绩。

(2)第7行声明整型数组num，并为数组元素赋初值，该整型数组共有8个元素。

(3)第9~17行输出数组里各元素的内容，并进行成绩汇总。

(4)第18行输出sum的结果即为总销售量。

4.3 多维数组

经过前面一维、二维数组的练习后不难发现，想要提高数组的维数，只要在声明数组的时候将索引与中括号再加一组即可，所以三维数组的声明为int A[][][]，而四维数组为int A[][][][]，以此类推。

使用多维数组时，输入、输出的方式和一维、二维相同，但是每多一维，嵌套循环的层数就必须多一层，所以维数越高，其复杂度也就越高。以三维数组为例，在声明数组时即赋初值，再将其元素值输出并计算总和。

范例 TestJava4_6.java

```
1.  //下面程序说明了三维数组的使用方法，输出数组的内容需要采用三重循环
2.  public class TestJava4_6
3.  {
4.    public static void main(String args[])
5.      {
6.        int i,j,k,sum=0;
7.        int A[][][]={{{5,1},{6,7}},{{9,4},{8,3}}};//声明数组并设置初值
```

```
        //三维数组的输出需要采用三层 for 循环方式输出
        for(i=0;i<A.length;i++)                    //输出数组内容并计算总和
           for(j=0;j<A[i].length;j++)
              for(k=0;k<A[j].length;k++)
                {
                 System.out.print("A["+i+"]["+j+"]["+k+"]=");
                 System.out.println(A[i][j][k]);
                 sum+=A[i][j][k];
                 }
          System.out.println("sum="+sum);
    }
}
```

输出结果如图4-10所示。

图4-10 TestJava4_6.java的运行结果

由于使用的是三维数组，所以嵌套循环有3层。

4.4 方法

方法既可以简化程序的结构，也可以节省编写相同程序代码的时间，达到程序模块化的目的。在每一个类里出现的 main()即是一个方法。使用方法来编写程序代码有相当多的好处，它可简化程序代码、精简重复的程序流程，并把具有特定功能的程序代码独立出来，使程序的维护成本降低。

方法的语法格式如下。

格式4-9 声明方法，并定义其内容

```
返回值类型      方法名称(类型 参数1,类型 参数2, …)
{
程序语句;
return 表达式;
方法的主体
}
```

要特别注意的是，如果不需要传递参数到方法中，只要将括号写出，不必填入任何内容。此外，如果方法没有返回值，则 return 语句可以省略。

4.4.1 方法操作的简单范例

TestJava4_7.java 是一个简单的方法操作范例，它在显示器上先输出 19 个星号“*”，换行之后再输出“I Like Java！”这一行字符串，最后再输出 19 个星号。

范例 TestJava4_7.java

```
1.  //以下程序主要说明如何声明并使用一个方法
2.  public class TestJava4_7
3.  {
4.      public static void main(String args[])
5.      {
6.          star ();                    //调用 star () 方法
7.          System.out.println("I Like Java !");
8.          star ();                    //调用 star () 方法
9.      }
10.     public static void star()
11.     {
12.         for(int i=0;i<19;i++)       //star () 方法
13.         System.out.print("*");       //输出 19 个星号
14.         System.out.print("\n");      //换行
15.     }
16. }
```

输出结果如图4-11所示。

TestJava4_7.java 中声明了两个方法，分别为 main()方法与 star()方法。因为 main()方法是程序进入的起点，所以把调用 star()的程序代码编写在 main()里。在 main()的第 6 行调用 start() 方法，此时程序的运行流程便会进到第 11~15 行的 star()方法里执行。执行完毕后，程序返回 main()方法，继续运行第 7 行，输出“I Like Java！”字符串。第 8 行调用 sart()方法，程序再次进到第 11~15 行的 star()方法里运行。运行完后，返回 main()方法里，因 main()方法接下来已经没有程序代码可供执行，于是结束程序 TestJava4_7.java。

图4-11 TestJava4_7.java的运行结果

从本程序中，可以很清楚地看出，当调用方法时，程序会跳到被调用的方法里去运行，结束后则返回原调用处继续运行。在 TestJava4_7.java 中，调用与运行 star()方法的流程如图4-12所示。

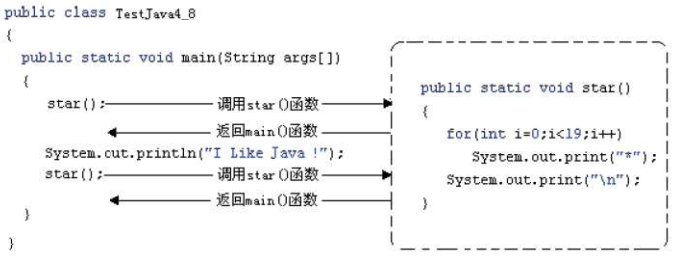

图4-12 调用与运行 star()方法的流程

在程序 TestJava4_7.java中 star()方法并没有任何返回值，所以 star()方法前面加上了一个 void 关键字。此外，因为 star()没有传递任何的参数，所以 star()方法的括号内保留空白即可。至于在 star() 方法之前要加上 static 关键字，这是因为 main()方法本身也声明成static，而在 static 方法内只

能访问到 static 成员变量（包括数据成员和方法成员）之故，因 star()方法被 main()方法所调用，自然也要把 star()声明成 static。如果还不了解 static 的真正用意也没有关系，本书将在后续的章节对 static 关键字做详尽的介绍。

4.4.2 方法的参数与返回值

如果方法有返回值，则必须在声明方法之前指定返回值的数据类型。同样，如果有参数要传递到方法内，则必须在方法的括号内填上该参数及其类型。TestJava4_8.java是用来说明方法的使用的另一个范例，它可以接收一个整数参数 n，输出 2×n 个星号后，返回整数 2×n。

范例 TestJava4_8.java

```
1.  //以下程序是关于方法的返回类型是整型的范例
2.  public class TestJava4_8
3.   {
4.      public static void main(String args[])
5.       {
6.         int num;
7.         num=star(7);      //输入 7 给 star(),并以 num 接收返回的数值
8.         System.out.println(num+" stars printed");
9.       }
10.     public static int star(int n)  //star() method
11.     {
12.        for(int i=1;i<=2*n;i++)
13.          System.out.print("*");     //输出 2*n 个星号
14.        System.out.print("\n");      //换行
15.        return 2*n;                  //返回整数 2*n
16.     }
17. }
```

输出结果如图4-13所示。

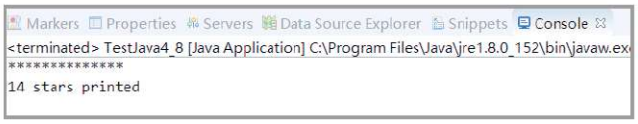

图4-13 TestJava4_8.java的运行结果

在TestJava4_8.java中，因star()传递整数值，所以第10行的声明要在star() 方法之前加上 int 关键字。此外，因要传入一个整数给 star()，所以 star()的括号内也要注明参数的名称与数据类型。

如果要传递一个参数，只要在方法的括号内填上所要传入的参数名称与类型即可。TestJava4_9.java 是一个关于计算长方形对角线长度的范例，其中 show_length()方法可接收长方形的宽与高，计算后返回对角线的长度。

范例 TestJava4_9.java

```
1.  //以下的程序说明了方法的使用
2.  public class TestJava4_9
3.   {
4.     public static void main(String args[])
5.       {
6.        double num;
```

```
7.        num=show_length(22,19); //输入 22 与 19 两个参数到 show_
                                  //length()里
8.        System.out.println("对角线长度 = "+num);
9.       }
10.    public static double show_length(int m, int n)
11.     {
12.       return Math.sqrt(m*m+n*n);     //返回对角线长度
13.    }
14. }
```

输出结果如图4-14所示。

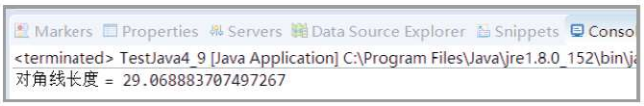

图4-14 TestJava4_9.java的运行结果

TestJava4_9.java的第7行调用show_length(22,19)，把整数22和19传入show_length()方法中。第12行则利用Math类里的sqrt()方法计算对角线长度。而sqrt(n)的作用是将参数n开根号。因为sqrt()的返回值是double类型，因此show_length()返回值也是double类型。

4.4.3 方法的重载

方法的重载就是在同一个类中允许同时存在一个以上的同名方法，只要它们的参数个数或类型不同即可。在这种情况下，该方法就叫被重载了，这个过程称为方法的重载。

范例 TestJava4_10.java

```
1. //以下程序说明了方法的重载操作
2. public class TestJava4_10
3.     {
4.     public static void main(String[] args)
5.     {
6.         int int_sum;
7.         double double_sum;
8.         int_sum = add(3,5);          //调用有两个参数的 add() 方法
9.         System.out.println("int_sum = add(3,5)的值是:"+int_sum);
10.        int_sum = add(3,5,6);        //调用有3个参数的 add() 方法
11.        System.out.println("int_sum = add(3,5,6)的值是:"+int_sum);
12.        double_sum = add(3.2,6.5); //传入的数值为 double 类型
13.        System.out.println("double_sum = add(3.2,6.5)的值是:
                              "+double_sum);
14.    }
15.    public static int add(int x,int y)
16.    {
17.        return x+y;
18.    }
19.    public static int add(int x,int y,int z)
20.    {
21.        return x+y+z;
22.    }
23.    public static double add(double x,double y)
```

```
24.     {
25.         return x+y;
26.     }
27. }
```

输出结果如图4-15所示。

可以发现 add 被重载了3次，但每个重载了的方法所能接收参数的个数和类型不同。

图4-15　TestJava4_10.java的运行结果

4.4.4 将数组传递到方法中

方法不只可以用来传递一般的变量，也可用来传递数组。本节将讲述在 Java 里是如何传递数组及如何处理方法的返回值是一维数组的问题。

1. 传递一维数组

要传递一维数组到方法里，只要指明传入的参数是一个数组即可。TestJava4_11.java是传递一维数组到 largest() 方法的一个范例，当 largest() 接收到此数组时，便会把数组的最大值输出。

范例 TestJava4_11.java

```
1.  //一维数组作为参数来传递，这里的一维数组采用静态方式赋值
2.      public class TestJava4_11
3.      {
4.      public static void main(String args[])
5.      {
6.          int score[]={7,3,8,19,6,22};      //声明一个一维数组 score
7.          largest(score);     //将一维数组 score 传入 largest() 方法中
8.      }
9.      public static void largest(int arr[])
10.     {
11.         int tmp=arr[0];
12.         for(int i=0;i<arr.length;i++)
13.         if(tmp<arr[i])
14.             tmp=arr[i];
15.         System.out.println("最大的数 = "+tmp);
16.     }
17. }
```

输出结果如图4-16所示。

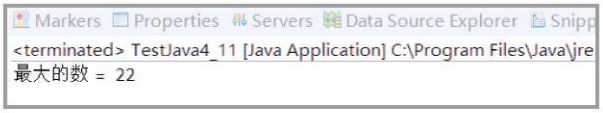

图4-16　TestJava4_11.java的运行结果

TestJava4_11.java的第 11~16 行声明 largest() 方法，并将一维数组作为该方法的参数。第 12~15行找出数组的最大值，并将它输出来。注意如果要传递数组到方法里，只要在方法内填上数组的名称即可，如本范例的第 7 行所示。

2. 传递二维数组

二维数组的传递与一维数组相当类似，只要在方法中声明传入的参数是一个二维数组即可。程序 TestJava4_12.java 是有关传递二维数组的一个范例，把二维数组 A 传递到print_mat()方法里，并在 print_mat()方法里把该数组值输出。

范例 TestJava4_12.java

```
1.  //以下程序说明了如何将一个二维数组作为参数传递到方法中
2.      public class TestJava4_12
3.      {
4.      public static void main(String args[])
5.      {
6.          int A[][]={{51,38,22,12,34},{72,64,19,31}};
7.          print_mat(A); //定义一个二维数组 A
8.      }
9.      public static void print_mat(int arr[][])//接收整数类型的二维数组
10.         {
11.             for(int i=0;i<arr.length;i++)
12.             {
13.                 for(int j=0;j<arr[i].length;j++)
14.                     System.out.print(arr[i][j]+" "); //输出数组值
15.                 System.out.print("\n");  //换行
16.             }
17.         }
18.     }
```

输出结果如图4-17所示。

TestJava4_12.java的第 9~17 行声明了print_mat()方法，它可接收二维数组，并利用两个 for 循环把数组的值输出来。注意可以利用.length 取出数组的行数或列数，如程序的第 11 行与第 13 行所示。

图4-17　TestJava4_12.java的运行结果

3. 返回数组的方法

如果方法返回整数，则必须在声明时在方法的前面加上 int 关键字。相反，如果返回的是一维的整型数组，则必须在方法的前面加上 int[]。若是返回二维的整型数组，则加上 int[][]，以此类推。

TestJava4_13.java 是返回二维数组的一个范例。将一个二维数组传入 add10()方法中，在add10()方法内将每一个元素加 10 之后返回它，最后在 main()里输出此数组。

范例 TestJava4_13.java

```
1.  //以下程序说明了方法中返回一个二维数组的实现过程
2.      public class TestJava4_13
3.      {
4.          public static void main(String args[])
5.          {
6.              int A[][]={{51,38,82,12,34},{72,64,19,31}};
```

```
                                        //定义二维数组
7.              int B[][]=new int[2][5];
8.              B=add10(A);          //调用 add10(),并把返回的值设给数组 B
9.              for(int i=0;i<B.length;i++)      //输出数组的内容
10.             {
11.                 for(int j=0;j<B[i].length;j++)
12.                     System.out.print(B[i][j]+" ");
13.                 System.out.print("\n");
14.             }
15.         }
16.         public static int[][] add10(int arr[][])
17.         {
18.             for(int i=0;i<arr.length;i++)
19.                 for(int j=0;j<arr[i].length;j++)
20.                     arr[i][j]+=10;     //将数组元素加 10
21.                 return arr;            //返回二维数组
22.         }
23.     }
```

输出结果如图4-18所示。

图4-18 TestJava4_13.java的运行结果

虽然 TestJava4_13.java的程序代码较长，但不难理解。第 16 行赋值 add10()是可接收二维数组，且返回类型是二维的整型数组。第 20行是完成了在循环内将数组元素值加 10 的操作，而运算之后的结果再由第 21 行的 return 语句返回。

【案例描述】

输入数组，最大的与第一个元素交换，最小的与最后一个元素交换，输出数组。

【技术要点】

设置两个变量max、min分别存放最大值和最小值，初始化时赋值为a[0]，利用循环把每次的最大值和最小值分别存放在max和min中，并保留相应的位置。

【代码与注释】

```
1.  import java.util.*;
2.  public class SwapMaxMin {
3.  public static void main(String[] args) {
4.      int N = 8;
5.      int[] a = new int [N];
6.      Scanner s = new Scanner(System.in);
7.      int idx1 = 0, idx2 = 0;
8.      System.out.println("请输入8个整数:");
```

```
9.     for(int i=0; i<N; i++) {
10.     a[i] = s.nextInt();
11. }
12.    System.out.println("你输入的数组为:");
13.    for(int i=0; i<N; i++) {
14.      System.out.print(a[i] + " ");
15.    }
16.    int max =a[0], min = a[0];
17.    for(int i=0; i<N; i++) {
18.     if(a[i] > max) {
19.      max = a[i];
20.      idx1 = i;
21.     }
22.     if(a[i] < min) {
23.      min = a[i];
24.      idx2 = i;
25.     }
26.    }
27.    if(idx1 != 0) {
28.     int temp = a[0];
29.     a[0] = a[idx1];
30.     a[idx1] = temp;
31.    }
32.     if(idx2 != N-1) {
33.     int temp = a[N-1];
34.     a[N-1] = a[idx2];
35.     a[idx2] = temp;
36.    }
37.    System.out.println("\n交换后的数组为:");
38.    for(int i=0; i<N; i++) {
39.     System.out.print(a[i] + " ");
40.    }
41. }
42. }
```

【运行结果】

案例SwapMaxMin.java的运行结果如图4-19所示。

图4-19 SwapMaxMin.java的运行结果

【相关知识】

Java中交换两个变量值的3种方法。

第一种：生成第三方变量（多占内存）。

```
1.  int temp=a;    //生成第三方临时变量temp，将a的值赋给它
```

```
2.  a=b;              //将b的值赋给a
3.  b=temp;           //将临时变量temp的值赋给b
```

这样就完成了a、b之间值的交换。

第二种：不生成第三方变量，数值相加减（可能有精度损失）。

```
1.  a=a+b; //将a+b的值赋给a
2.  b=a-b; //将a-b的值赋给b，注意这里的a是原来a+b的值，即这里就是把原来a的值赋
           //给了b
3.  a=a-b; //将a-b的值赋给a，注意这里的b是原来的a的值，即这里就是把原来b的值赋
           //给了a
```

第三种：使用位运算中的异或，两次异或不改变原值（a = a ^ b ^ b，一个数连续与另一个数进行两次运算结果还是这个数）。

```
1.  a=a^b;
2.  b=a^b; //注意这里a的值，此时右边的表达式为：( a^b )^b ，结果是a，赋予变量b;
3.  a=a^b; //这里这里a,b都不是原来的值了，此时右边的表达式为：a^( a^b )，结果是b，
           //赋予变量
```

实训1　将一个数组逆序输出

【实训目的】

掌握数组的应用。

【实训内容】

将一个数组逆序输出。

【代码与注释】

```
1.  import java.util.*;
2.  public class Inverted {
3.  public static void main(String[] args) {
4.      Scanner s=new Scanner(System.in);
5.      int a[]=new int[20];
6.      System.out.println("请输入多个正整数（输入-1表示结束）:");
7.      int i=0,j;
8.      do{
9.         a[i]=s.nextInt();
10.        i++;
11.     }while (a[i-1]!=-1);
12.     System.out.println("你输入的数组为:");
13.     for( j=0; j<i-1; j++) {
14.         System.out.print(a[j]+"    ");
15.     }
16.     System.out.println("\n数组逆序输出为:");
17.     for( j=i-2; j>=0; j=j-1) {
18.      System.out.print(a[j]+"    ");
```

```
    }
  }
}
```

【实训结果】

Inverted.java运行结果如图4-20所示。

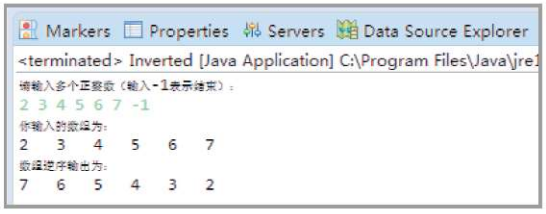

图4-20 Inverted.java的运行结果

【小贴士】

数组反转和逆序输出的思路如下。

（1）数组反转的思路是用0号元素和数组最后一个元素进行互换，然后分别继续互换到start(前一个元素下标)大于end(后一个元素下标)的时候停止互换，打印反转后的数组，这个思路可以应用到逆序输出。

```
public class Inverted {
    public static void main(String[] args) {
        int temp;
        int arr[]= {11,2,22,33,44,8};
        for(int i=0;i<arr.length/2;i++) {
            temp=arr[i];
            arr[i]=arr[arr.length-i-1];
            arr[arr.length-i-1]=temp;
        }
        for(int i=0;i<arr.length;i++)
            System.out.print(arr[i]+" ");

    }
}
```

（2）逆序输出直接从后面倒过来输出即可（可以提高一点点效率）。

【实训目的】

掌握数组的综合应用。

【实训内容】

在超市购物之后，收银台需要打印购物清单。每次输入一个商品的名称、数量及价格，结束时输入-1，最后应该计算出所有商品的总价格。再输入一个顾客支付的金额，可以打印出相关数据，包括商品的名称、数量、单价、总额、顾客所付金额及应该找给顾客的金额。

【代码与注释】

```
import java.util.Scanner;
public class PrintStoreReceipts {
 public static void main(String[] args) {
```

```
4.          Scanner scanner=new Scanner(System.in);
5.          int[] commodityCode,quantity;
6.          double unitPrice[],total=0,paid,returnMoney;
7.          commodityCode=new int[10];
8.          quantity=new int[10];
9.          unitPrice=new double[10];
10.         int i;
11.         for(i=0;i<10;i++) {
12.             System.out.print("请输入商品的名称：蔬菜-1，肉类-2，食品-3，
日常用品-4，退出-——1:");
13.             commodityCode[i]=scanner.nextInt();
14.             if(commodityCode[i]==-1)
15.              break;
16.             System.out.print("请输入商品的数量:");
17.             quantity[i]=scanner.nextInt();
18.             System.out.print("请输入商品的单价:");
19.             unitPrice[i]=scanner.nextDouble();
20.         }
21.         System.out.println("购物小票\n商品名称\t数量\t单价");
22.         for(int j=0;j<i;j++) {
23.              switch(commodityCode[j]) {
24.                  case 1:System.out.println("蔬菜\t\t"+
                            quantity[j]+"\t"+unitPrice[j]);
25.                  break;
26.                  case 2:System.out.println("肉类\t\t"+
                            quantity[j]+"\t"+unitPrice[j]);
27.                  break;
28.                  case 3:System.out.println("食品\t\t"+
                            quantity[j]+"\t"+unitPrice[j]);
29.                  break;
30.                  case 4:System.out.println("日常用品\t"+
                            quantity[j]+"\t"+unitPrice[j]);
31.                  break;
32.                  default:break;
33.                  }
34.              total=total+unitPrice[j]*quantity[j];
35.         }
36.         System.out.println("商品金额总计:"+total);
37.         System.out.print("请输入顾客所付金额:");
38.         paid=scanner.nextDouble();
39.         returnMoney=paid-total;
40.         System.out.println("商品金额总计:"+total);
41.         System.out.println("顾客所付金额:"+paid);
42.         System.out.println("应找金额:"+returnMoney);
43.  }
44. }
```

【实训结果】

PrintStoreReceipts.java 结果如图 4-21 所示。

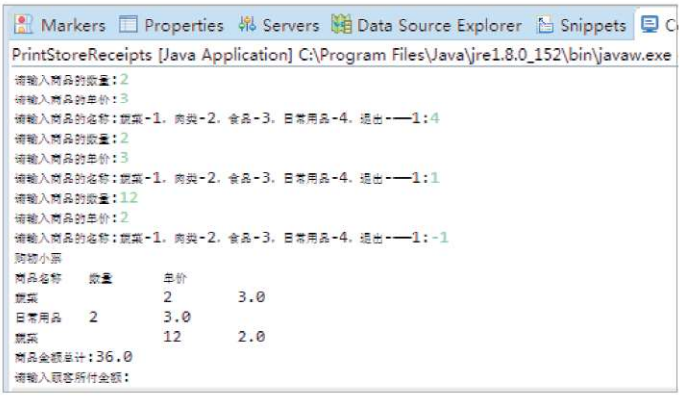

图4-21 PrintStoreReceipts.java的运行结果

【小贴士】

java.util.Arrays 类能方便地操作数组，它提供的所有方法都是静态的。使用时导入包import java.util.Arrays。该包具有以下功能(部分)。

输出数组：通过toString方法不需要循环遍历直接输出数组。

对数组排序：通过 sort 方法将数组升序排列。

复制数组：通过copyOf方法将一个数组的值赋值给另一个数组。

勇攀科技高峰的“追光者”

“物有甘苦，尝之者识；道有夷险，履之者知。”在中华民族伟大复兴的征程上，一代又一代科学家心系祖国和人民，不畏艰难，无私奉献，为科学技术进步、人民生活改善、中华民族发展作出了重大贡献。从“两弹一星”到载人航天，从“天眼”问天到万米深潜……一项项举世瞩目的科技成就，不断书写着我国科技自立自强的恢宏篇章。

在一代代科技工作者的接续奋斗中，以“爱国、创新、求实、奉献、协同、育人”为内涵的科学家精神得以涵育发展、发扬光大。这一精神，是伟大建党精神的时代观照，是大德、公德、品德在科技界的生动写照，正在全社会汇聚起正能量、振奋起精气神，助力科学的种子撒向更广阔的祖国大地。

党的十八大以来，习近平总书记高度重视科技发展、关心科技工作者，围绕推进科技创新、建设世界科技强国，对科技工作者寄予厚望。近年来，广大科技工作者牢记习近平总书记的殷殷嘱托，以国之昌盛、民之福祉为己任，立足岗位，拼搏奋进，助力今日之中国信息畅通、高铁飞驰、“神舟”飞天、“嫦娥”探月、“蛟龙”入水、“天河”运转……处处释放着科技发展的无限生机和澎湃动力。新时代新征程上，广大科技工作者要继续肩负起党和人民的重托，满怀信心，迎难而上，为实现高水平科技自立自强贡献智慧和力量！

（资料来源：王碧薇，《勇攀科技高峰的“追光者”——科学家精神背后的故事》，《党建》，2024-01-02，有改动）

操作题

1. 有一个已经排好序的数组，现输入一个数，要求按原来的规律将它插入数组中。
2. 对10个数进行排序。
3. 求100之内的素数。
4. 打印杨辉三角形。