普通高等教育“十五”国家级规划教材配套参考书

Student Solutions and Lab Manual to
Java Programming and Cases

Java程序设计与案例习题解答与实验指导

刘宝林　主编

高等教育出版社
Higher Education Press

普通高等教育“十五”国家级规划教材配套参考书

Student Solutions and Lab Manual to Java Programming and Cases

Java 程序设计与案例习题解答与实验指导

刘宝林　主编
胡　博　谢锋波　编

高等教育出版社

内容简介

本书为普通高等教育“十五”国家级规划教材刘宝林主编《Java 程序设计与案例》的配套习题解答与实验指导。全书由刘宝林主编。

全书共分 2 部分:第 1 部分是习题解答,给出了每一章的学习目标和习题解答。第 2 部分是实验指导,共给出了 5 个综合案例实验,每个案例包括实验目的、案例分析、程序设计(或系统设计)、代码实现(或构建)和运行结果(或应用程序部署)5 个部分,读者通过这些案例可将各知识点结合起来,达到学以致用的目的。本书附录对本书综合案例中涉及的 J2ME、J2EE 及设计模式等分别进行了简单的介绍,并对配书光盘的使用进行了说明。本书所附光盘包括《Java 程序设计与案例》教材的配套电子教案、教材中所有案例和本书所有综合案例的源代码及 Java 开发工具。

本书所有代码均在 J2SDK 1.4.1 平台下调试通过。

本书可作为高等院校计算机专业或非计算机专业、各类成人教育学院 Java 程序设计课程、计算机水平考试培训教材的教学辅导书,也可供同等程度的读者及计算机应用开发人员自学使用。

图书在版编目(CIP)数据

Java 程序设计与案例习题解答与实验指导/刘宝林主编.—北京:高等教育出版社,2005.7

ISBN 7-04-017251-8

Ⅰ.J… Ⅱ.刘… Ⅲ.Java语言-程序设计-高等学校-自学参考资料 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(2005)第 051633 号

策划编辑 何新权 **责任编辑** 俞丽莎 **封面设计** 于文燕 **责任绘图** 朱 静
版式设计 胡志萍 **责任校对** 俞声佳 **责任印制** 韩 刚

出版发行	高等教育出版社	**购书热线**	010-58581118
社 址	北京市西城区德外大街 4 号	**免费咨询**	800-810-0598
邮政编码	100011	**网 址**	http://www.hep.edu.cn
总 机	010-58581000		http://www.hep.com.cn
经 销	北京蓝色畅想图书发行有限公司	**网上订购**	http://www.landraco.com
印 刷	北京原创阳光印业有限公司		http://www.landraco.com.cn
开 本	787 × 1092 1/16	**版 次**	2005 年 7 月第 1 版
印 张	21.5	**印 次**	2005 年 7 月第 1 次印刷
字 数	480 000	**定 价**	29.90 元(含光盘)

本书如有缺页、倒页、脱页等质量问题,请到所购图书销售部门联系调换。

物料号 17251-00

前　言

本书为普通高等教育“十五”国家级规划教材刘宝林主编《Java 程序设计与案例》的配套习题解答与实验指导。

全书共分 2 部分：

第 1 部分是习题解答。每一章内容分为学习目标和习题解答。“学习目标”中归纳、总结了这一章应该掌握的内容，便于读者有的放矢，总结提高；“习题解答”中详细给出了教材中全部习题的参考答案，包括文字解释、程序代码、图示等。

第 2 部分是实验指导。共给出了 5 个综合案例实验，每个案例包括实验目的、案例分析、程序设计(或系统设计)、代码实现(或构建)和运行结果(或应用程序部署)5 个部分，读者通过这些案例可将各知识点结合起来，达到学以致用的目的。综合案例实验 1 通过演示常用的 Swing 组件，使学生熟悉面向对象的编程方法，体会继承和重载的应用，掌握 Java 图形用户界面的设计，学会使用 JFC Swing 常用组件及事件驱动模型；综合案例实验 2 通过编写一个 Java 版本的“连连看”游戏，使学生学习 Java 桌面程序的设计，学会灵活运用 Java 组件和使用 Java 进行算法实现；综合案例实验 3 通过创建一个网上的虚拟社区或在线交友平台，使学生掌握如何开发基于 JSP 的 Web 应用，学习使用apache ant 构建 Java 应用，学会在项目中使用开放源代码软件(OSS)和部署 Web 应用；综合案例实验 4 通过展示 J2EE 应用的一个演示程序 Pet Store，使学生掌握 J2EE 应用程序的开发流程，熟悉常见的 J2EE 应用程序框架和 MVC 设计模式，了解 O /R Mapping 工作机制；综合案例实验 5 通过介绍“微型电子辞典”系统，使学生掌握 J2ME 的基本概念和编程，学会使用模拟器调试程序和应用 JSP /Servlet，了解如何下载 J2ME 程序到手机。教师在指导这些综合实验时可根据实际情况对实验内容进行取舍。

本书附录中对综合案例中涉及的 J2ME、J2EE 及设计模式等分别进行了简单介绍。同时还对配书光盘的使用进行了说明。本书所附光盘包括《Java 程序设计与案例》教材的配套电子教案、教材中所有案例和本书所有综合案例的源代码及 Java 开发工具。

本书所有代码均在 J2SDK 1.4.1 平台下调试通过。

本书内容与教材紧密配合，习题和案例的编排本着循序渐进、由浅入深的原则，学生学习后可做到举一反三。习题部分既有简单的概念题和基本语法验证题，也有较难的综合性题目，在巩固学生对基本知识和基本方法掌握的基础上，拓宽学生思路，提高学生的综合应用能力。实验指导部分强调理论与实践的结合，注重提高读者利用面向对象技术和 Java 语言解决实际问题的能力。读者通过边学边练，在有限的学时内，可掌握面向对象程序设计的基本知识、基本方法和技巧。

本书可作为高等院校计算机专业或非计算机专业、各类成人教育学院程序设计课程、计算机水平考试培训教材的教学辅导书，也可供同等程度读者及计算机应用开发人员自学使用。

本书由刘宝林主编。第1章～第12章由刘宝林、胡博、谢锋波编写，综合实验指导及附录由谢锋波、胡博编写，常玉、陈晨参加了本书习题的编写工作，孟威、王彦杰、刘俊玲参加了本书程序的调试，全书最后由刘宝林统一修改定稿。特别感谢王行言教授认真审阅了全书，并提出了许多宝贵的建议。

由于作者水平所限，加之时间紧迫，书中难免有欠妥之处，恳请专家、读者批评指正。

刘宝林

2005年3月于清华园

目　录

第1部分　习题解答

第2部分　实验指导

第1部分　习题解答

第1章　Java概述

学习目标

- 了解Java语言的发展历史和Java平台的应用划分
- 了解Java语言的特点及其实际应用领域
- 了解Java开发环境及开发工具，掌握如何进行开发环境设置及如何编译和运行Java程序
- 熟悉几种不同类型的Java程序

习题解答

1．安装系统，熟悉J2SDK基本命令。

答：略。

2．Java语言的主要特点有哪些？

答：Java语言的主要特点如下：

(1)(simple)易学，自动内存管理，简化重载，去掉指针及C++中一些不是绝对必要的功能。Java语言与C++语言的风格极为相似，但却比C++语言简单得多，Java语言去掉了C++语言中容易引发程序错误的地方。

(2)(object-oriented)纯面向对象语言，程序代码以类的形式组织，由类来定义对象的各种状态和行为。具备面向对象的四大特点：抽象、封装、继承和多态。

(3)(distributed)丰富的网络编程功能，轻松处理TCP/IP，通过URL(统一资源定位器)访问远地资源；字节码可来自网络。

(4)(interpreted)Java是解释型的，但Java通过预先将源代码编译为接近于机器指令的字节码，有效地克服了传统解释型语言的性能瓶颈，同时又保持了解释型语言的可移植性。Java解释器能直接在任何机器上执行Java字节码。

(5)(robust)静、动态检查，排除出现错误的条件。异常处理，取消指针，内存保护。Java语言系统仔细检查对内存的每次访问，确认它是合法的，以免引起任何问题。如果出现某种意外，系统不会崩溃，而是把该异常抛弃，取消指针，从而杜绝了对内存的非法访问。

(6) (secure) 适用于网络/分布式运算环境,确保建立无病毒且不会被侵入的系统。内存分配及布局由 Java 运行系统决定。字节码加密传输,客户端校验。Java 程序分 Application 应用程序和 Applet 两种。Application 在本地执行,而 Applet 可在网上发布,但需要通过浏览器执行,为保证从远程下载的 Applet 不会对用户造成危害,Java 引入了砂盒(sandbox)安全模型,限制小程序访问本地资源。

(7) (architecture-neutral) 让 Java 应用程序能够在网络上任何地方执行,字节代码实现平台无关性,完全统一的语言版本实现无关性,访问底层操作系统功能的扩展类库使程序不依赖于具体系统。Java 语言源程序被编译成一种高层次的、与机器无关的以及结构中立的字节码语言,该格式语言可以在 Java 虚拟机上运行,只要有 Java 语言运行系统的机器都能执行这种中间代码。

(8) (portable)Java 本身环境可移植。Java 程序可在配备了 Java 解释器和运行环境的任何机器上运行,这成为 Java 软件便于移植的良好基础。

(9) (high-performance)将字节码转换成目标代码。Java 开发者设计了 just in time 编译器(也叫代码生成器),这种编译器可以在运行时把 Java 的字节码翻译成特定的机器代码,从而提高了性能。

(10) (multi-threaded)支持多任务。在语言级嵌入了对并发控制的功能——多线程控制,大大简化了多线程应用程序的开发,使应用程序可以并行执行,在一个程序里可同时执行多个小任务,同步机制保证了对共享数据的正确操作。

(11) (dynamic) 可动态增加和修改类库内容,是面向对象设计的延伸。Java 的基本组成单元是类,而 Java 的类又是运行时动态装载的,因此,可以在分布环境中动态地保持应用程序和类库的一致性,以便更好地适应时刻变化的环境。Java 不必因程序库的更新而重新编译程序。Java 语言的设计使它更适合于一个不断发展的环境,在类库中可以自由地加入新的方法和实例变量而不会影响用户程序的执行,同时允许程序动态地装入运行过程中所需要的类。

3. Java 语言主要有哪些方面的应用?

答: 基于 Java 语言的诸多特点,Java 语言主要应用于以下几个方面:

(1) 用于不同机型、不同操作系统计算机之间的数据交换和通信,完成协调控制、综合管理等功能。

(2) 用于可视化图形软件和多媒体软件的设计。

(3) 用于计算机交互软件的设计与开发。

(4) 为 Internet 网络用户设计生动、活泼、带动画的主页。

4. 试述 Java 语言、Java 虚拟机和 Java 平台三者之间的关系。

答: Java 语言就是编写 Java 应用程序的时候所用的编程语言,是一种面向对象的编程语言,语法与 C 语言类似。设计者在保证 Java 语言强大功能的同时,避免了像其他面向对象语言那种过于复杂的语法(比如 C++)。而且对开发人员而言,使用 Java 语言编写强鲁棒性的代码也同样方便快捷。出色的设计,使得 Java 语言广受赞誉。

编译 Java 语言的代码后,可以得到一种中间结果,称为字节码,字节码可以在一种机器语言

解释环境上解释执行，这种类似 CPU 结构的运行环境称为 Java 虚拟机（简称为 JVM[1]）或 Java 解释器[2]。Java 语言的初始设计目标，就是保证 Java 程序可以在所有拥有 Java 虚拟机的环境中运行。目前大部分常见的操作系统上都拥有对应的 Java 虚拟机，如 Windows、Solaris、Apple MacOS 等，除了这些计算机操作系统以外，Java 虚拟机也存在于诸如电视机顶盒、移动外设等领域。

与这二者不同的是，Java 平台是指 Java 类库的集合，有的时候，Java 平台也和 Java 运行环境和核心 Java API 联系在一起，所有的 Java 程序都建立在一些预定义的 Java 类库基础之上。

5．编写一个 Application，在屏幕上显示如下的信息：

```
**********************************
        Welcome <你的姓名>
**********************************
```

答：源程序 N1Code.java：

```
package c1;

import java.io.*;

public class N1Code
{
    public static void main(String args[])
    {
System.out.println("**********************************");
System.out.println("        Welcome <你的姓名>");
System.out.println("**********************************");
    }
}
```

6．将上题改写为 Applet 小程序。

答：源程序 N2Code.java：

```
package c1;

import java.applet.Applet;
import java.awt.Graphics;

public class N2Code extends Applet
{
    public void paint(Graphics g)
    {
```

```
        g.drawString("********************************", 10, 20);
        g.drawString("        Welcome <你的姓名> ", 10, 25);
        g.drawString("********************************", 10, 30);
    }
}
```

相应的 HTML 文件 N2Code.html 的源代码为：

```
<HTML>
<HEAD>
<TITLE>WelcomeApplet </TITLE>
</HEAD>
<BODY>
<applet code = N2Code.class width = 170 height = 150>
</applet>
</BODY>
</HTML>
```

第 2 章　Java 语言基础

学习目标

- 熟练掌握 Java 语言的数据类型、常量、变量、表达式及流程控制语句
- 掌握数组的定义、建立及其使用方法
- 熟悉 Java 一般程序的结构

习题解答

1. 判断下面的标识符在 Java 中是否合法：

a. itsID　　b. 8you　　c. _isID　　d. player-name

答：合法的是 a、c 和 d。

2. Java 的变量有哪些类型？变量如何定义？如何初始化？

答：Java 的变量有两大类型：基本类型和复合类型。

基本类型包括：数值类型、布尔类型和字符类型。

复合类型包括：数组、字符串和类类型。

和大多数高级语言一样，Java 语言也要求变量先定义后使用。定义形式如下：

```
type variable;
```

前面为变量类型，后面为变量名。变量类型为 Java 语言中给出的类型；变量名是由字母、数字、下画线或“$”符号组成，且必须以字母、下画线或“$”开头。定义时变量名要遵循见名知意的原则。一般变量名的第一个字母小写，若由两个以上的单词构成，则从第二个单词开始，每一个单词的第一个字母要大写。

Java 语言中，在定义变量的同时可以对其初始化。

基本数据类型的变量，如：

```
int i=1;
```

复合数据类型的变量，如：

数组：`int[] arry={1,2,3,4,5};`

字符串：`String str="Tsinghua";`

类类型：`Employee a=new Employee();`

3. 什么时候需要强制类型转换？举一个与主教材 2.1.2 中不同的例子。

答：将占用内存较多的数据类型的变量或表达式转换成占用内存较少的数据类型时，需要使用强制类型转换，其使用格式如下：

(数据类型)变量名或表达式

例如:

```
byte MyByte=10;
int MyInteger= -1;
```

把 MyInteger 的值赋给 MyByte 时,必须进行强制类型转换:

```
MyByte= (byte)MyInteger;
```

4. Java 的控制语句有几种类型? 举例说明其使用方法。

答: Java 的控制语句有 3 种类型:分支语句、循环语句和跳转语句。

分支语句包括 if-else 语句和 switch-case 语句。

循环语句包括 for 语句、while 语句和 do-while 语句。

跳转语句包括 break 语句、continue 语句和 return 语句。

(1) if-else 语句的一般形式为:

```
if (boolean-expression)
    {statement or block
    }
else
    { statement or block
    }
```

例如:

```
if (a > b)
    {
      System.out.println("a is bigger than b");
    }
else
    {
      System.out.println("a is not bigger than b");
    }
```

(2) switch-case 语句,即开关语句,其功能是根据表达式的值在多个 case 引导的多分支语句中选择一个来执行。

switch-case 语句的一般形式如下:

```
switch(expression1){
        case expression2:
             statements;
             break;
        case expression2:
             statements;
             break;
```

```
    ...
    default:
        statements;
        break;
}
```

其中 switch 子句中的表达式的值必须是整数或者是字符类型；case 子句使用的判断值必须是常数，而不能是变量或者表达式；“break”用于结束整个分支；default 子句在没有任何判断值与 switch 表达式的值相等的情况下执行。

例如：

```
switch (c) {
    case 'a':
        System.out.println("The value of c is : 'a' ");
        break;
    case 'b':
        System.out.println("The value of c is : 'b' ");
        break;
    case 'c':
        System.out.println("The value of c is : 'c' ");
        break;
    default:
        System.out.println("c has value : '"+ c + "'");
        break;
}
```

for 语句是循环语句中功能最强的语句，其一般形式如下：

```
for(expression1; boolean-expression2; expression3){
    statement or block;
}
```

“expression1”是用来进行循环变量初始化等工作的，它只在 for 循环刚开始的时候执行一次；在初始化后，会判断“boolean-expression2”的值，如果为真，则执行一次循环体内的程序块，否则退出循环；执行一次循环体内的程序块后，会执行“expression3”，一般是对循环变量的值进行更新，之后再判断“boolean-expression2”的值，如果为真，则再执行一次循环体内的程序块，否则退出循环。

例如：

```
for (i=1;i <= 10; i++){
    System.out.println("Have you finished up it ?");
}
```

(3) while 语句的一般形式如下：

```
while (boolean-expression)
    statement or block;
```

while 循环在执行时，先判断条件表达式的值，如为真，则执行一次程序块，然后再一次进行判断；否则结束循环。

例如：

```
int i = 0;
while (i <= 10){
    System.out.println("Have you finished up it ?");
    i++;
}
```

(4) do-while 语句的一般形式如下：

```
do
    statement or block;
while (boolean-expression);
```

和 while 循环不同的是，do-while 循环先执行一次程序块，再判断表达式的值，如为真，则执行一次程序块，然后再一次进行判断；否则结束循环。

例如：

```
int i = 0;
do {
    System.out.println("Have you finished up it ?");
    i++;
} while (i <= 10);
```

(5) continue 只能用在循环结构中，用来终止一次循环操作，直接执行下一次循环。continue 语句又分为不带标号的 continue 语句和带标号的 continue 语句。

不带标号的 continue 语句表示提前结束本次循环，即跳过后面的循环体语句，回到循环的条件测试部分继续执行。

例如：

```
import java.io.*;
public class NoNumberContinue
{
    public static void main(String args[])
    {
        int i;
        for(i=1;i<18; i++)
        {
            System.out.print("*");
            if(i%4!=0)
```

```
                continue;
                System.out.print();
            }
        }
    }
```

该程序用于显示“*”号，每显示 4 个“*”号后，自动换行。

带标号的 continue 语句采用“label:”的形式定义标号，它跳过循环剩余语句，直接进入标号所指的循环体的下一轮循环。这里需注意的是，continue 语句不是 goto 语句，标号不能指向与本循环并列的其他循环，也不可指向非循环语句。

例如：

```
import java.io.*;
public class prime
{
    public static void main(String args[])
    {
        First_Loop:
        for(int i=2,k=0;i<100;i++)
        {
          for(int j=2;j<=Math.sqrt(i);j++)
          {
            if(i%j==0)
            continue First_Loop;
          }
        System.out.print(i+"\t");
        k++;
        if(k%5==0)
          System.out.println();
        }
    }
}
```

(6) break 语句则是终止整个循环。break 语句也分为不带标号的 break 语句和带标号的 break 语句。

不带标号 break 语句从循环体内跳出至后面语句，结束当前循环体。循环嵌套时，break 语句只跳出当前循环体。

例如：

```
import java.io.*;
public class NoNumberBreak
```

```
{
    public static void main(String args[])
    {
        int i,j,k=0;
        for(i=1; i<=15; i++)
          for(j=1; j<15; j++)
          {
            if(i==5)
              continue;
            if(j>5)
              break;
            k++;
          }
        System.out.print("k="+k);
    }
}
```

带标号的 break 语句表示跳出标号标志的循环体。

例如：

```
import java.io.*;
public class labeled {
    public static void main(String[] args) {
        int i = 0;

        outer:                                  //标号定义 outer
        while(true) {                           //外层循环
            prt("Outer loop top");

            while(true) {                       //内层循环
                prt("inner loop top");
                i++;
                prt("i = "+ i);
                if(i == 1) continue;            //跳过本次循环,开始另一次循环
                if(i == 3) continue outer;      //跳过本次循环,进入 outer 所指的循环
                if(i == 5) break;               //跳出内层循环体
                if(i == 7) break outer;         //跳出 outer 所指的外层循环体
                prt("inner loop tail");
            }
            prt("Outer loop tail");
```