

高等学校计算机公共基础课规划教材



Java 程序设计 习题解析与实验指导

杨厚群 伍 强 主 编

王业统 陈 静 副主编



中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

高等学校计算机公共基础课规划教材

Java 程序设计习题解析与实验指导

主 编 杨厚群 伍 强

副主编 王业统 陈 静

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

本书是应广大读者的要求, 为了配合《Java 程序设计》(杨厚群主编)一书的学习而编写的。本书分为三大部分, 第1部分是主教材各章基本知识点的讲解、典型题分析和习题讲解, 通过对习题进行讲解和分析, 读者可以进一步加深对Java程序设计基本概念的理解, 对基本技术的运用, 对基本知识的掌握。第2部分是Java实验指导。第3部分是两个附录, 其中Java集成开发环境介绍紧贴主教材给出了丰富的实验, 并对实验做出分析与提示, 方便读者上机操作。

本书适合作为高等学校计算机及相关专业“Java 程序设计”课程的辅助教材, 也可作为学习Java语言的实验指导教材。

图书在版编目(CIP)数据

Java 程序设计习题解析与实验指导 / 杨厚群, 伍强主编. —北京: 中国铁道出版社, 2009. 10
高等学校计算机公共基础课规划教材
ISBN 978-7-113-10662-1

I. J… II. ①杨…②伍… III. JAVA 语言—程序设计—高等学校—教学参考资料 IV. TP312

中国版本图书馆CIP数据核字(2009)第184463号

书 名: Java 程序设计习题解析与实验指导
作 者: 杨厚群 伍 强 主编

策划编辑: 严晓舟 杨 勇
责任编辑: 秦绪好
编辑助理: 李 旸
封面设计: 付 巍
责任印制: 李 佳

编辑部电话: (010) 63583215
责任校对: 李庆祥
封面制作: 李 路

出版发行: 中国铁道出版社(北京市宣武区右安门西街8号 邮政编码: 100054)
印 刷: 北京市兴顺印刷厂
版 次: 2009年11月第1版 2009年11月第1次印刷
开 本: 787mm×1092mm 1/16 印张: 10 字数: 239千
印 数: 4000
书 号: ISBN 978-7-113-10662-1/TP·3602
定 价: 16.00元

版权所有 侵权必究

本书封面贴有中国铁道出版社激光防伪标签, 无标签者不得销售

凡购买铁道版的图书, 如有缺页、倒页、脱页者, 请与本社计算机图书批销部调换。

本书是《Java 程序设计》一书的学习指导和补充教材。应读者要求,为了配合“Java 程序设计”课程的学习,针对主教材的各章内容提供习题和答案,希望读者通过练习进一步加深对 Java 程序设计基本概念的理解,对基本技术的运用,对基本知识的掌握。

全书分三大部分:

第 1 部分是主教材各章基本知识点的讲解、典型题分析和习题。基础知识讲解内容包括需要了解和需要理解掌握的两个层面的知识点。读者在学习时应按照学习指导所给出的重点来学习,从而达到事半功倍的效果。在典型题分析部分给出每一章的典型题分析详解,可帮助读者了解掌握一些重要的解题思路和方法。在此提醒读者切忌死记硬背习题答案,而是要在理解本书的解答后给出自己的答案。其实,习题的解答并不一定是唯一的,简答题和程序设计题都是如此。通过练习习题,复习和掌握主教材的内容,进一步加深对 Java 程序设计的理解,快速提高分析问题和解决问题的能力。

第 2 部分是 Java 程序设计实验指导。在本部分进一步加强了实验和设计等教学环节,根据主教材章节的内容安排了 12 个实验,并给出了实验的详细要求和部分实验分析和提示,供读者参考。

第 3 部分是两个附录。附录 A 介绍 Java 集成开发环境,附录 B 是第 1 部分中各章节习题的参考答案。

本书由杨厚群、伍强任主编,由王业统和陈静任副主编,王业统和陈静对范例进行了验证并解答了习题。

由于编者水平有限,本书难免有错误和不足之处,诚恳希望读者不吝指正,以便在今后的工作中进一步改进和提高。

编 者

2009 年 5 月

第 1 部分 知识点与习题

第 1 章	Java 概述	1
1.1	本章知识重点	1
1.2	典型题分析	1
1.3	习题	1
第 2 章	Java 基本编程结构	3
2.1	本章知识重点	3
2.2	典型题分析	7
2.3	习题	12
第 3 章	字符串和数组	21
3.1	本章知识重点	21
3.2	典型题分析	22
3.3	习题	23
第 4 章	对象和类	26
4.1	本章知识重点	26
4.2	典型题分析	30
4.3	习题	32
第 5 章	继承与多态	35
5.1	本章知识重点	35
5.2	典型题分析	37
5.3	习题	40
第 6 章	异常处理	43
6.1	本章知识重点	43
6.2	典型题分析	44
6.3	习题	45
第 7 章	图形编程	47
7.1	本章知识重点	47
7.2	典型题分析	48
7.3	习题	51
第 8 章	Java Swing 与事件处理	54
8.1	本章知识重点	54
8.2	典型题分析	56
8.3	习题	60

第 9 章 Applet 基础	63
9.1 本章知识重点	63
9.2 典型题分析	64
9.3 习题	66
第 10 章 多线程	69
10.1 本章知识重点	69
10.2 典型题分析	71
10.3 习题	74
第 11 章 输入/输出流及文件	78
11.1 本章知识重点	78
11.2 典型题分析	79
11.3 习题	83
第 12 章 Java 的网络编程	87
12.1 本章知识重点	87
12.2 典型题分析	90
12.3 习题	95

第 2 部分 实验指导

实验 1 熟悉 Java 程序开发环境	97
1.1 实验目的	97
1.2 实验内容	97
实验 2 Java 基本编程结构	99
2.1 实验目的	99
2.2 实验内容	99
2.3 分析与提示	99
实验 3 字符串和数组	101
3.1 实验目的	101
3.2 实验内容	101
3.3 分析与提示	102
实验 4 对象和类	103
4.1 实验目的	103
4.2 实验内容	103
4.3 分析与提示	104
实验 5 继承和多态	105
5.1 实验目的	105
5.2 实验内容	105
5.3 分析与提示	106

实验 6 异常处理	107
6.1 实验目的	107
6.2 实验内容	107
6.3 分析与提示	108
实验 7 图形编程	109
7.1 实验目的	109
7.2 实验内容	109
7.3 分析与提示	109
实验 8 Java Swing 与事件处理	110
8.1 实验目的	110
8.2 实验内容	110
8.3 分析与提示	110
实验 9 Applet 基础	113
9.1 实验目的	113
9.2 实验内容	113
9.3 分析与提示	114
实验 10 多线程编程	117
10.1 实验目的	117
10.2 实验内容	117
10.3 分析与提示	118
实验 11 Java 输入与输出	121
11.1 实验目的	121
11.2 实验内容	121
11.3 分析与提示	121
实验 12 网络编程	123
12.1 实验目的	123
12.2 实验内容	123
12.3 分析与提示	124

第 3 部分 附录

附录 A Java 集成开发环境	128
A.1 DOS 环境 Java 2 SDK	129
A.1.1 建立 Java 开发环境	129
A.1.2 JDK 开发环境中常用工具介绍	131
A.2 小型集成视窗环境 JCreator 介绍	132
A.3 大型集成开发环境 Eclipse	133
A.4 Java 程序举例	136

A.4.1 简单的 Java 应用程序	136
A.4.2 简单的 Applet 小程序	137
附录 B 实验教程习题参考答案	139
参考文献	150

第 1 部分 知识点与习题

第 1 章 // Java 概述

1.1 本章知识重点

Java 语言是由美国 Sun 公司推出的新一代面向对象的程序设计语言，特别适用于 Internet 应用程序的开发。学习本章重点在于掌握 Java 语言的发展简史，Java 的特点，Java 在 Internet 中的应用，安装 J2SDK 开发环境和如何使用集成开发环境。

1.2 典型题分析

简答题

Java 程序分为哪两个大类？它们之间有哪些区别？

答：根据结构和运行环境的不同，Java 程序可以分为两类：Java Application 和 Java Applet。

简单地说，Java Application 是完整的程序，需要独立的解释器来解释运行，可以独立运行，以 main 方法作为入口；Java Applet 则是嵌在用 HTML 编写的 Web 页面中的非独立程序，由 Web 浏览器内部包含的 Java 解释器来解释运行，它不能独立运行，可以使用 AppletViewer 或其他支持 Java 的浏览器运行。Java Application 和 Java Applet 各自使用的场合也不相同，但不论是哪种 Java 程序，都用 .java 为扩展名的文件存放。

1.3 习 题

一、选择题

1. Java 采用（ ）机制来删除对象。
 - A. Java 的 delete 保留字
 - B. 析构函数
 - C. 无用内存收集机制
 - D. Java 不删除对象（这个工作由开发人员负责）
2. Java 语言的特点是（ ）。
 - A. 面向对象、支持网络和解释型
 - B. 健壮、安全、结构中立和可移植
 - C. 高性能、多线程和动态
 - D. 以上选项都是

- #### D. appletviewer test.html

按照书中 Java 程序的编写、编译和运行步骤,试编写一个简单的 Java 程序在屏幕上输出自己的名字。

第 2 章 // Java 基本编程结构

2.1 本章知识重点

1. Java 符号集

符号是构成程序的基本单位。Java 语言采用 Unicode 字符集,在这种字符集中,每个字符用 2 个字节即 16 位表示。Java 语言中所使用的符号按词法可分为如下 5 类:

(1) 标识符 (identifiers): 程序员对程序中的各个元素加以命名时使用的命名记号称为标识符。Java 语言中,标识符是以字母、下画线 (_)、美元符 (\$) 开始的一个字符序列,后面可以跟字母、下画线、美元符、数字。例如, identifier、userName、User_Name、_sys_val、\$change 为合法的标识符,而 2mail、room#、class 为非法的标识符。

(2) 关键字 (keyword): 具有专门的意义和用途,不能当做一般的标识符使用,这些标识符称为关键字,也称为保留字,如 abstract、continue、for、new、switch、boolean、default 等。Java 语言中的关键字均用小写字母表示。

(3) 运算符 (operands)。

① 双目运算符 (见表 2-1):

表 2-1 双目运算符

运 算 符	运 算	例 子	功 能
+	加	a+b	求 a 与 b 的和
-	减	a-b	求 a 与 b 的差
*	乘	a*b	求 a 与 b 的积
/	除	a/b	求 a 与 b 的商
%	取余	a%b	求 a 与 b 相除的余数

② 单目运算符 (见表 2-2):

表 2-2 单目运算符

运 算 符	运 算	例 子	功 能
++	自增	a++或++a	a=a+1
--	自减	a--或--a	a=a-1
-	取反	-a	

③ 关系运算符 (见表 2-3):

表 2-3 关系运算符

运 算 符	运 算
==	等于
!=	不等于
>	大于
<	小于
>=	大于等于

④ 逻辑运算符 (见表 2-4):

表 2-4 逻辑运算符

运 算 符	运 算
&	非简单与
	非简单或
!	取反
^	异和
&&	简单与

⑤ 位运算符 (见表 2-5):

表 2-5 位运算符

运 算 符	运 算	例	规 则
~	位反	~x	按位取反
>>	右移	x>>a	按位右移 a 位
<<	左移	x<<a	按位左移 a 位
>>>	不带符号的右移	x>>>a	按位右移 a 位, 左边空位填 0

⑥ 其他运算符: 三目运算符?:; 圆括号与方括号()、[]; 对象运算符 instanceof。

(4) 常量 (literals): 程序运行过程中不变的量, 用保留字 final 来实现, 如 final int NUM=100;。

(5) 注解: Java 的注解有两种, 即行注解 (//), 其中//称为单行注解; 注释块 (以/*开始, 到*/结束), /*...*/称为多行注解; 文档注解, 以单斜线加两个星形标记 (/**) 开头, 并以一个星形标记加单斜线 (*/) 结束。

2. 数据类型、常量和变量

(1) 数据类型。Java 的数据类型可划分为基本数据类型和复杂数据类型, 基本数据类型包括布尔型、字符型、整型、浮点型。数据类型是对数据的取值范围和允许对这些数据施加的操作的封装。

(2) 常量。Java 中的常量, 按其数据类型来划分, 有整数型常量、浮点型常量、布尔型常量、字符型常量和字符串常量等 5 种。

(3) 变量及其声明。变量是在程序的运行过程中其值可以被改变的量。变量名是用户自己定义的标识符, 而变量的值则是这个变量在某一时刻的取值。任何计算机程序设计语言在使用变量

之前，必须要先声明变量。声明变量，就是指定变量的数据类型。

3. 表达式

表达式是用运算符把操作数连接起来表达某种运算或含义的式子。根据表达式中所使用的运算符和运算结果的不同，可以将表达式分为以下几种：算术表达式、赋值表达式、关系表达式、逻辑表达式、位运算。

4. 运算符的优先级

运算符的优先级决定了表达式中不同运算执行的先后次序。优先级高的先进行运算，优先级低的后进行运算。在优先级相同的情况下，由结合性决定运算的顺序。

最基本的规律是：域和分组运算的优先级最高，接下来依次是单目运算、双目运算、三目运算，赋值运算的优先级最低。

5. 程序的三种基本结构

流程控制语句是用来控制程序中各语句执行顺序的语句，是程序中非常关键和基本的部分。流程控制语句可以把单个语句组合成有意义的，能完成一定功能的小逻辑模块，能否熟练地运行流程控制语句很大程度上影响程序的质量。最主要的流程控制方式是结构化程序设计中的三种基本结构，如图 2-1 所示。

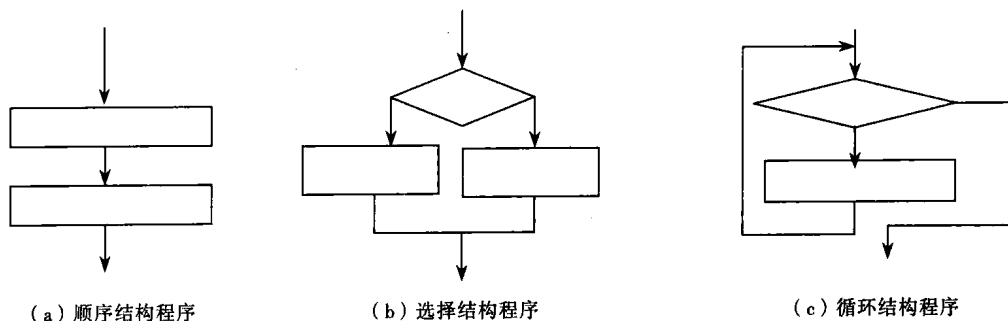


图 2-1 三种基本结构

选择结构程序可进一步分为三种不同的形式：双分支选择结构程序、单分支选择结构程序、多分支选择结构程序。使用 if 语句可以构造这三种形式的选择结构程序。

① if...else 语句的基本形式：

if(布尔表达式)

语句区块 1

else

语句区块 2

② 省略了 else 的 if 语句：

if(布尔表达式)

语句区块

③ if 语句的嵌套：

if 语句的常见嵌套形式如下：

else if(布尔表达式 2) 语句区块 2

else if(布尔表达式 3) 语句区块 3

...

else 语句区块 n+1

6. switch 语句

switch 语句根据表达式的值来决定执行许多不同操作中的某一个操作，即使用 switch 语句是构造多分支选择结构程序的简明形式。其语句格式如下：

```
switch
{
    case 值 1: 语句区块 1;break;           //分支 1
    case 值 2: 语句区块 2;break;           //分支 2
    ...
    case 值 n: 语句区块 n;break;           //分支 n
    [default: 语句区块 n+1;]               //分支 n+1
}
```

7. 构造循环结构的 while 语句

构造循环结构的 while 语句的格式如下：

```
while(布尔表达式)
{
    循环体语句区块
}
```

8. 构造循环结构的 do...while 语句

Do...while 语句的一般形式为：

```
do{
    循环体语句区块
}while(布尔表达式)
```

9. 构造循环结构的 for 语句

for 语句的一般形式为：

```
for(初值表达式; 布尔表达式; 循环过程表达式)
{
    循环体语句区块
}
```

for 语句执行过程如下：

- (1) 求解初值表达式的值。
- (2) 求解布尔表达式的值，若值为真，则执行循环体语句区块，然后再执行第 (3) 步；若值为假，则跳出循环语句。
- (3) 求解循环过程表达式后，转去执行第 (2) 步。

10. 循环控制要点

控制循环主要有两种办法，一是用计数器控制循环，二是用标号控制循环。

大多数循环结构程序是利用计数器的原理来控制的。设计用计数器控制循环的程序，需要把握下面几个要点：

- (1) 循环控制变量（或循环计数器）的名字，即循环变量名。
- (2) 循环控制变量的初始值。
- (3) 每执行一次循环时，循环控制变量的增量（或减量）。
- (4) 测试循环控制变量的终值条件（即是否继续进行循环）。

通过仔细分析三种循环控制语句可以看出：在用三种不同的循环语句编写的程序中，都具有

上述四个要点的内容,其中以 for 语句最为典型,它把所有这些构件都放在了 for 语句头中。图 2-2 明确地指出了这几个要求。

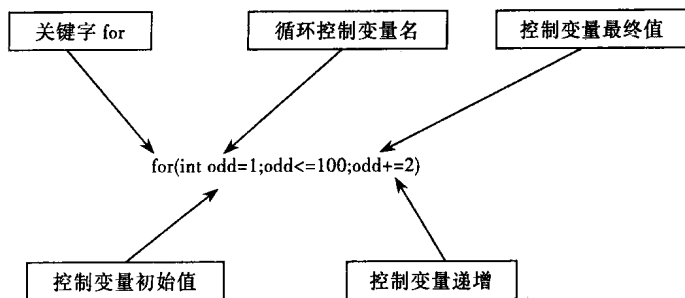


图 2-2 一个典型 for 语句头的组成构件

11. 三种循环语句的比较

Java 系统提供了三种循环语句,一般情况下,这三种循环语句是可以相互替代的,尤其是对于那些确切地知道所需执行次数的循环。然而,由于 for 语句头中包含了控制循环所需要的各个构件,因此对于同样的问题,使用 for 循环编写的程序最简洁清晰。如果读者将求自然数 1~10 之和的问题分别用三种循环结构写出,就可以清楚地看到这一点。然而,对于那些只知道某些语句要反复执行多次(至少执行一次),但不知道确切的执行次数时,使用 do...while 循环可以使程序更清晰;而对于那种某些语句可能要反复执行多次,也可能一次都不执行的问题,当然使用 while 循环最好。

12. break 语句和 continue 语句

break 语句虽然可以独立使用,但通常主要用于 switch 结构和循环结构中控制程序的执行流程转移。

continue 语句只能用于循环结构中。在循环结构中的 continue 语句的作用是当程序执行到 continue 语句时,程序会回到循环的入口处执行下一次循环,而使循环体内写在 continue 语句后的语句不执行。

当程序中有嵌套的多层循环时,若想从内循环跳到外循环,可以使用带标号的 continue lab 语句,此时应在外循环的入口语句的前方加上标号。

2.2 典型题分析

一、简答题

1. 说明在数据类型转换中,什么是隐式类型转换?什么是显式类型转换?

答:强制类型转换是指一个表达式中出现的各种变量或常量的数据类型不同时,所进行的数据类型转换。Java 程序中的类型转换可分为显式和隐式转换两种形式。

(1) 隐式类型转换。对于由双目运算中的算术运算符组成的表达式,一般要求运算符两边的两个操作数的类型一致,如果两者的类型不一致,则系统会自动转换为较高(即取值范围较大)的类型,这便是隐式数据类型转换。例如:

```
int x=3;
double y;
```

y=x;

执行这段程序后 y 的值等于 3.0。

(2) 显式类型转换。隐式类型转换只能由较低类型向较高类型转换, 若需要由较高类型向较低类型转换则需要显式类型转换。显式类型转换需要人为地在表达式前面指明所需要转换的类型, 系统将按这一要求把某种类型强制性地转换为指定的类型, 其一般形式如下:

(<类型名>) <需要进行类型转换的表达式>

例如:

```
double x=1.5,y=3.7;
```

```
int z;
```

```
z=(int)(x+y)%2/4;
```

执行这段程序后 z 的值等于 0。

2. while 和 do...while 循环语句的区别。

答: while 语句是先判断布尔表达式, 如果布尔表达式的值为 true 就执行循环体, 否则不执行; do...while 语句是先执行一次循环体后判断布尔表达式, 如果布尔表达式的值为 true 就继续循环体, 否则不执行。所以, while 语句的循环体可能一次都不执行, 而 do...while 语句的循环体最少执行一次。例如:

```
i=10;
sum=0;
while(i<10)
{
    sum+=i;
    i++;
}
```

这段程序的循环体一次都不执行, 所以执行后 sum=0。

```
i=10;
sum=0;
do{
    sum+=i;
    i++;
}while(i<10)
```

这段程序的循环体执行一次, 所以执行后 sum=1。

3. break 语句和 continue 语句的区别。

答: break 语句主要用于 switch 结构和循环结构中, 控制程序的执行流程转移。break 语句用在 switch 语句中, 其作用是强制退出 switch 结构, 执行 switch 结构后的语句; 在单层循环结构的循环体中, 其作用是强制退出循环结构。若程序中有内外两重循环, 而 break 语句写在内循环中, 则执行 break 语句只能退出内循环, 而不能退出外循环。若想要退出外循环, 可使用带标号的 break 语句。continue 语句用于循环结构中, 作用是当程序执行到 continue 语句时, 程序会回到循环的入口处执行下一次循环, 而使循环体内写在 continue 语句后的语句不执行。例如:

```
sum=0;
for(int i=1;i<=5;i++)
{
    if(i==4)
        break;
    sum+=i;
}
```


这段程序当 $i=4$ 时, 退出循环, 所以 $\text{sum}=1+2+3=6$ 。

```
sum=0;
for(int i=1;i<=5;i++)
{
    if(i==4)
        continue;
    sum+=i;
}
```

这段程序当 $i=4$ 时, 退出本轮循环进入下轮循环, 所以 $\text{sum}=1+2+3+5=11$ 。

二、编程题

1. 编写一个 Java 程序, 求出一元二次方程 $ax^2+bx+c=0$ 的两个根并打印出来。

解: 当 $a=0$ 时, 方程式不是一元二次方程; 当 $a \neq 0$ 且 b^2-4ac (判别式) $=0$ 时, 一元二次方程有两个相等的实根; 当 $a \neq 0$ 且 b^2-4ac (判别式) >0 时, 一元二次方程有两个不相等的实根; 当 $a \neq 0$ 且 b^2-4ac (判别式) <0 时, 一元二次方程有两个不相等的虚根。

```
public class Roots
{
    public static void main(String[] args)
    {
        double a,b,c;                //方程二次项系数、一次项系数、常数项
        double x1=0,x2=0;             //方程的两根
        double disc;                  //判别式
        double realpart=0,imagpart=0; //复数根的实虚部
        a=3;
        b=4;
        c=5;
        if(a==0)                      //判断是否是一元二次方程
            System.out.println("此方程不是一元二次方程");
        else
        {
            disc=b*b-4*a*c;            //求判别式的值
            if(disc==0)
                System.out.println("方程有两个相等的根: "+(-b/(2*a)));
            else if(disc>0)
            {
                x1=(-b+Math.sqrt(disc))/(2*a);
                x2=(-b-Math.sqrt(disc))/(2*a);
                System.out.print("方程有两个不相等的根: ");
                System.out.println("x1="+x1+",x2="+x2);
            }
            else
            {
                realpart=-b/(2*a);
                imagpart=Math.sqrt(-disc)/(2*a);
                System.out.print("方程有两个复数根: ");
                System.out.println("x1="+realpart+"+"+imagpart+"i");
                System.out.println("x2="+realpart+"-"+imagpart+"i");
            }
        }
    }
}
```