经全国中小学教材审定委员会2004年初审通过

普通高中课程标准实验教科书

信息技术 · 选修1

算法与程序设计

SUANFA YU CHENGXU SHEJI

祝智庭 主编

中国地图出版社

经全国中小学教材审定委员会2004年初审通过

普通高中课程标准实验教科书

信息技术 · 选修1

算法与程序设计

SUANFA YU CHENGXU SHEJI

祝智庭　主编

中国地图出版社

本套教科书主编　祝智庭
本套教科书副主编　刘观武　任友群　高淑印
本册教科书主编　祝智庭
本册教科书副主编　高淑印

责任编辑　沈万君　兰大鹏
美术编辑　张　萌
审　　校　李红梅
复　　审　陈书香
审　　订　李俊生

经全国中小学教材审定委员会 2004 年初审通过
普通高中课程标准实验教科书

书　　名　信息技术·选修 1
算法与程序设计
主　　编　祝智庭

出　　版　中国地图出版社
社　　址　北京市白纸坊西街 3 号
邮 政 编 码　100054
电　　话　010-83060930
地图教学网　www.ditu.cn
印　　刷
发　　行
成 品 规 格　210mm × 297mm
印　　张　10
版　　次　2005 年 1 月第 1 版　2010 年 6 月第 2 版
印　　次　2019 年 6 月　第 18 次印刷

书　　号　ISBN 978-7-5031-5372-3
定　　价　元（含光盘定价：5.00 元）
批 准 文 号　举报电话：12358

编写说明

本套教科书根据教育部《普通高中技术课程标准（实验）·信息技术》编写，经全国中小学教材审定委员会2004年初审通过，供高中阶段学习使用。本套教科书共分六册：必修模块为《信息技术基础》，选修模块依次为《算法与程序设计》《多媒体技术应用》《网络技术应用》《数据管理技术》和《人工智能初步》。

整套教科书以“知识引领、活动穿插，任务引领、知识渗透，工具支持、资源配套，评估跟进、形式多样”为编写思路，从解决学生日常生活和学习中的实际问题入手，运用信息获取、加工、管理、表达与交流的基本方法，在主题活动、探究性学习等多种形式的学习过程中逐步提升学生的信息素养，从而实现知识与技能、过程与方法、情感态度与价值观三个方面的培养目标。为了方便学习和创作，便于过程性评价，教科书引入了基于网络环境的“电子学习档案袋”。每册教科书附配套学习光盘（CD-ROM），从课本资料、视频教程、学生范例、技术扩展和学生自测系统等几方面为课程学习提供帮助。我们还建立服务网站，及时更新和拓展教学资源，为教师和学生提供教与学的支持，为便于教师运用电子学习档案袋、电子作品和在线考试平台等多种评价方式对学生的学习效果进行测评，网站还特别提供了相关软件和辅助资源。

自初审通过，本套教科书已在多个新课改实验区使用多年。为了进一步提高教科书的质量和普适性，在充分调研的基础上，2010年3月，我们组织人员对本册教科书进行了修订，力图更完善地呈现教科书的科学性、通用性和前瞻性。

本册教科书为选修1模块，供36学时使用。

本套教科书由华东师范大学教授、博士生导师祝智庭任主编，特级教师刘观武和华东师范大学教授任友群博士、天津市中小学教育教学研究室高淑印任副主编。

本册教科书主编祝智庭，副主编高淑印，编写者方文祺、齐国英、黄福铭、刘观武。

参与本册教科书修订的人员有高淑印、郭莉、何平、张茹桂、腾伟。

欢迎广大师生通过电子邮件（infotech@sinomaps.com）与我们交流，提出意见和要求，指出差错或不足，共同推动信息技术课程和教材的建设。

天津市中小学教育教学研究室

中 国 地 图 出 版 社

2010年6月

前言

当今社会的信息化发展已进入一个以移动互联网、智能设备、云计算和大数据为特征的全新阶段，新一代信息技术正在以前所未有的方式和速度渗透并影响着各个领域、行业，推动着人类社会进入万物互联和普适计算的新时代。当“程序驱动”的数字化工具改变着我们的生活、学习和工作，甚至成为人们身体的一部分时，我们不仅要具备操作这些技术工具的技能，还要从深层次理解这些技术工具，知道它们的工作方法和应用流程，处理好人与技术工具的关系，即发展计算思维，能够通过计算来解决问题。

计算与算法之间、程序与计算机之间，是唇齿相依的关系。计算机离不开程序的控制，程序的核心是算法的设计。

实际上算法的起源比电子计算机的出现要早很多。当同学们欣赏动画片、游戏等计算机实现的三维动画效果时，你们可知道这里面浸透着科学家上百年的算法研究成果？

当今指纹、虹膜、DNA 等生物特征识别技术的广泛应用，金融电子化与消费的增长，网络加密与反加密的斗争，对月球、火星的探测……你可知道这些新生事物的核心包括算法设计吗？

算法与程序设计造就了今天计算机的“智慧”和“能耐”，而创造这些神奇算法的人们，必将在算法的天地里，用程序代码编织出梦幻般的未来。

现在，让我们来共同探索这个神奇的算法与程序设计的世界，一起来感受人类智慧的辉煌吧！

目录

第一单元·走进编程

“授人以鱼不如授人以渔”。现在，掌握一门编程语言已不再令人感到神秘和高不可攀。但是，要想掌握一种解题方法，却不是每个人都能轻易做到的。人们编写程序就是为了解决问题，只有通过预先分析问题来确定必须达到的目标，才有希望找到解决问题的正确方案。

在刚踏入编程这个门槛时，我们先来了解计算机能解决哪些问题，并在经历计算机解决问题的基本过程中，掌握用算法去解决实际问题的基本方法，最终学会把复杂的问题转化为计算机能处理的程序。

第一节　解决问题的一般办法

日常生活中我们会遇到各式各样的问题，随着计算机技术的发展，大多数问题都可以使用计算机来解决。其中，有些问题可以使用现成的计算机工具软件解决，有些问题则需要编写特定的计算机程序才能解决。本节我们将学习解决问题的一般方法，特别要学会辨别哪些问题需要编写计算机程序来解决，并经历计算机解决问题的基本过程。

我们在解决问题时一般采取的步骤是：通过向老师或父母请教，上网或去图书馆查资料，初步形成自己的观点，然后设计解决方案并验证或反复验证观点，最终得到结论。即要经历猜想、验证和得出结论的过程。

21 世纪，信息技术为我们的世界带来了巨大的变革，计算机成为解决问题的重要工具。利用计算机解决问题的一般过程通常是先分析问题，确定解决问题的方案，根据方案来设计解决问题的具体步骤，然后交给计算机执行，最终解决问题。

让我们一起来经历利用计算机解决问题的过程吧！

一　分析问题

首先提出一个问题。学校的校医经常要对学生的体检表进行统计，分析出全校每个年级男、女同学的身高、体重及视力等身体状况。我们仅就其中一项指标来提出问题，如：全班同学谁最高？

分析问题首先要明确问题的目标和条件。该问题中目标为求出“全班同学谁的身高最高”，条件是已知全班每一个同学的身高。

其次要理解问题，即要搞清楚问题涉及了哪些方面的知识。该问题是对体检表中的身高数值进行比较。如果仅比较几个数的大小，直接观察就可以得出结果。但是，在比较大量数据的时候，就很难用直接观察的方法，需要考虑使用工具（如计算机）来解决。这时，就需要了解计算机处理数据（如比较数的大小）方面的相关知识。

1　提出能用计算机解决的问题。

现实生活中，我们经常会遇到各种各样的问题。例如，选择最近的上学路线，解数学计算题，填报升学志愿书时选择最满意的学校等等。请大家找出一些问题，在小组内讨论一下，把适用于计算机解决的问题提出来，填写在下面：

问题 1：学校期末考试后各科成绩统计：统计总分、平均分等项目。

问题 2：计算 10000 以内的奇数和。

问题 3：_____

问题 4：_____

问题 5：_____

二　确定方案

解决方案指采用什么工具、采取什么方法来解决问题。人们通常将问题的解决方案分为两大类：一类是通过若干个步骤就能得出问题结论的，这类问题的解决方案叫算法方案；另一类是不能通过若干个步骤直截了当地得出结论，而是需要反复验证才能解决的，这类问题的解决方案通常叫作启发式方案。在解决复杂的问题时，人们通常使用启发式方案。

计算机作为解决问题的重要工具，可以解决很多类型的问题。我们把能使用计算机解决的问题分为三类：

第一类：使用现有的工具软件可以解决的问题。

大多数需要用计算机解决的问题都属于这一类，例如，上网查询“百慕大三角区”的资料，编辑、打印班报，制作班级主页，收发邮件等。

解决这类问题的办法就是使用相应的计算机应用软件，如网络浏览器、文档编辑软件、主页制作软件、邮件收发软件等。

第二类：现成的工具软件解决不了，需要编写程序并通过执行程序才能解决的问题。

例如，计算 10000 以内的奇数和，查询某年的生肖，画 100 个同心圆，统计保龄球选手每人在每一局的积分等。

这类问题一般是：

◆计算型，指需要进行数学计算的问题；

◆逻辑型，指需要进行逻辑处理的问题；

◆需要反复执行一组计算型或逻辑型指令的问题。

这类问题主要使用算法方案解决，需要设计出解决问题的若干个具体步骤并编写程序。对于人们来说，手工处理这些问题既烦琐又耗时，但是用计算机处理起来就非常简单了。难点在于怎么设计具体的解决步骤，并用程序实现。

第三类：现成的工具软件不能直接解决，也需要编写程序，但不能通过既定步骤直接解决的问题。

例如，通过专家系统进行植物识别、农作物病虫害诊断；通过机器翻译对文字进行中英文互译；通过远程医疗网站进行诊病；人机对弈；等等。

这类问题不能由计算机通过既定步骤直接做出回答，具体问题需要具体分析，需要用启发式方案来解决。处理这类问题要利用计算机技术领域的人工智能技术。人工智能可以让计算机建立自己的知识库并学会人类的思维方式，从而使计算机解决问题的能力在某些方面和人类相差无几。

“全班同学谁最高”这个问题的解决方案可能有多种，我们试着列出一种：

让计算机来比较全班同学的身高值，找出最大值，即可知道全班同学谁最高。

此方案经过若干个步骤后能直接得出结论，并且不需要反复验证，由此可以判定该方

案属于算法方案，可以通过编写计算机特定的程序来解决。

2 列出解决问题的方案。

请大家把前面提出的问题进行归类，找出相对集中的一些问题，然后从简单问题入手，共同探讨解决方法，并把解决问题的方案列出来。考虑这些解决方案是属于哪一类解决方案，适合用计算机的哪一类工具软件来解决，能否通过有限的步骤得出问题的结论，是否需要反复验证。

问题 1 的解决方案：可以用计算机数据处理软件（如 Excel），进行成绩统计。这属于第一类能够利用现有的工具软件解决的问题。

问题 2 的解决方案：由计算机求出 10 000 以内的所有奇数，再将所有的奇数求和。此方案经过若干步骤后能直接得出结论，但需要通过编写计算机程序来解决，属于第二类问题。

问题 3 的解决方案：

问题 4 的解决方案：

问题 5 的解决方案：

三 设计步骤

现在我们根据方案来设计具体的解决步骤，即进行算法设计。因为考虑到是用计算机解决问题，所以这些步骤必须适用于计算机执行。

本例解决方案的关键是对身高值进行比较，找出最大值，据此可以列出如下步骤：

步骤 1：输入第一个同学的身高值；

步骤 2：输入下一个同学的身高值；

步骤 3：比较两个同学的身高值；

步骤 4：选出较大值；

步骤 5：再输入下一个同学的身高值；

步骤 6：用这个同学的身高值和上述较大值进行比较；

步骤 7：再选出较大值；

重复步骤 5 至步骤 7，直到输入最后一个同学的身高值，比较后选出较大值，该较大值即为全班同学中最高同学的身高值。

四 设计程序

算法方案通常都要通过编写特定的计算机程序来实现，即用计算机能够理解的语言（程序设计语言）将算法表达成程序，得出最终结果，这个过程就是设计程序。

问题分析是算法设计的基础，算法设计又是设计程序的基础。选择一种什么样的程序设计语言来解决问题并不是最重要的，关键是要把握算法与程序设计的思想。

例如，“全班同学谁最高”的问题，按上述算法编写的程序运行结果如图 1-1-1 所示。

图1-1-1　“全班同学谁最高”程序运行结果

1．对课堂上提出的问题进行筛选，找出适合用计算机解决的问题，列出其解决方案及具体的实施步骤。

2．打开配套光盘“sc\unit1”，执行小程序“判断闰年 .exe”，执行结果如图 1-1-2 所示。请回答下列问题：

(1) 这个程序解决了什么问题？

(2) 此问题如果不用计算机编程解决，还有其他办法吗？请写出解决步骤。

图1-1-2　判断闰年

第二节　解决问题的算法设计

本节我们将通过生活中的一个实例，学习如何设计算法并使用自然语言、伪代码和流程图等方法描述算法。

算法（Algorithm）就是解决问题的步骤序列。对问题进行算法描述之前，通常要先分析问题，设计相应的算法，然后使用自然语言、伪代码或流程图等描述算法，将解决问题的思路表达出来。

一　提出问题

乘出租车已成为城市生活中一件很平常的事。出租车收费的主要依据是所行驶的里程数和计费标准，地区不同、车型不同，计费标准也不相同。现有两种不同类型的出租车，其计费标准分别为：

甲车 3 千米起步，价格 10 元，3 千米以上（含）每千米为 2 元；乙车也是 3 千米起步，价格 8 元，3 千米以上（含）每千米为 2.2 元。

你能比较一下乘坐哪种车省钱吗？

如果用计算机编写程序来解决这个问题，需要经历问题分析、算法设计和程序实现三个过程。

二　分析问题

一般情况下问题分析应包括以下三个方面的内容。

1. 说明描述问题的任何假设

在一个问题中，假设就是为了方便设计而假定的认为是正确的描述。例如，我们假设不考虑外在的因素，如等候时间造成的计费问题，只考虑相同里程数的费用。

2. 已知信息

已知信息就是解决问题所需要的所有的已知条件。在描述问题时，要用“已知……”给出来。例如，已知两种出租车的计费标准和里程数。

3. 目标求解

目标求解就是如何解决问题，还要具体说明问题解决后如何输出结果。例如，让计算机输出乘坐哪种车省钱。

综合这三方面的内容，就可以进行完整的问题描述。问题描述的格式不唯一，可以用文字描述、表格描述或图形描述等，只要把问题表述清晰即可。

对出租车的计费问题可以这样描述：比较两种出租车的计费情况，假设没有等候时间；已知其计费标准和里程数；通过编程计算分别乘坐两种车的费用，并输出乘坐哪种车省钱。也可以使用表格形式描述问题，如表 1-2-1。

表1-2-1 出租车计费问题的描述

假 设	已 知	求 解
1. 甲、乙两种出租车 2. 无等候时间 3. 相同里程，费用低的省钱	1. 甲车起步价 10 元，以后每千米 2 元 2. 乙车起步价 8 元，以后每千米 2.2 元 3. 里程数为 n 千米	1. 乘甲车的费用（P1） 2. 乘乙车的费用（P2） 3. 比较 P1 与 P2，如果： P1 < P2，输出“n 千米时，乘甲车省钱” P1 > P2，输出“n 千米时，乘乙车省钱” P1=P2，输出“n 千米时，两车费用相同”

你认为哪种描述更明确呢？如果你有更好的描述方法，请提出来供大家参考。

三　设计算法

设计算法是计算机解决问题的重要环节。算法一般具有可执行性、确定性、有穷性等特征，即算法的每一个步骤都是可以执行的，按照这个步骤一步步执行就能够得出正确的结论，并且能够终止。

1. 算法的结构

（1）输入数据

输入数据就是输入已知信息。在设计算法时，首先应具体说明计算机如何输入这些信息。

（2）处理数据

处理数据就是列出计算机执行操作的指令序列。应该详细说明如何处理输入的数据，例如，根据里程数分别计算乘坐两种车的费用。

（3）输出结果

算法的最后还要描述计算机如何输出信息。

2. 算法的表示

表示算法的工具有自然语言、伪代码、流程图、结构化流程图（又称框图，即 N-S 图）等等。这些工具不是程序设计语言，不能直接被计算机执行，使用它们的目的就是把算法思想表达出来。

（1）自然语言

用自然语言描述算法就是使用人们能读懂的简短语句对算法的步骤进行描述。例如，乘出租车一题的算法，用自然语言描述如下：

①输入数据：

输入乘甲车的费用 P1 为起步价 10 元
输入甲车 3 千米以后每千米的费用 X1 为 2 元
输入乘乙车的费用 P2 为起步价 8 元
输入乙车 3 千米以后每千米的费用 X2 为 2.2 元
输入里程数 n

②处理数据：

如果 n 小于 3，则乘甲车的费用 P1 为 10 元，乘乙车的费用 P2 为 8 元
如果 n 大于等于 3，则乘甲车的费用为 P1+X1 × (n−3+1)
乘乙车的费用为 P2+X2 × (n−3+1)

③输出结果：

如果 P1<P2，输出“n 千米时，乘甲车省钱”
如果 P1>P2，输出“n 千米时，乘乙车省钱”
如果 P1=P2，输出“n 千米时，两车费用相同”

（2）伪代码

用伪代码描述算法就是采用一种类似于程序设计语言的代码来描述算法。伪代码没有统一的规定，只要定义合理，没有矛盾就可以。用伪代码表示的格式如下：

```
输入：.....
输出：.....
指令：.....
      .....
      .....
      .....
      .....
      .....
```

本书中定义了 4 种伪代码的基本指令：赋值指令、循环指令、条件指令和输出指令。

◆赋值指令

一般格式：助记符←表达式；

其中，助记符是对数据命名的符号，表达式是常量、助记符或是对助记符与常量进行运算的式子，箭头号“←”称为赋值号，每条指令以分号结束。

语义：对表达式进行计算，并将这个值赋给助记符，使助记符具有表达式的值。

例 1：P1 ← 10;

表示把 10 元起步价赋给 P1。

例 2：P1 ← P1+X1*（n−3+1）；

表示先计算赋值号右边的表达式 P1+X1*（n−3+1）的值，再把这个计算结果赋给 P1，这里“*”表示乘号。

◆输出指令

一般格式：输出（表达式）；

语义：输出表达式的值。表达式也包括表示文字信息的字符串，字符串要用西文双引号括起来。如果连续输出多项内容，各项之间用逗号分隔。

例如，输出“n 千米时，乘车费用相同”，用输出指令表示为：

输出（"n 千米时，两车费用相同"）；

◆条件指令

一般格式：

```
if（条件表达式）
{
    指令序列 1
}
else
{
    指令序列 2
}
```

语义：若“条件表达式”成立，则执行“指令序列 1”；否则（即“条件表达式”不成立）执行“指令序列 2”。如果指令序列中只有一条指令，则指令序列头尾的花括号可以省略，简写为“if（条件表达式）指令 1;”；若“指令序列 2”中无指令，那么“else”及其后边的花括号均省略。

条件表达式是用关系运算符来比较两个量的大小的，例如 x>3，其中“>”是关系运算符“大于”。x>3 的含义是：若 x 的值比 3 大，则表达式 x>3 的值为 true（真）；否则其值为 false（假）。另外，两个或多个简单的条件表达式还可以组合成复杂的条件表达式，这时就要用逻辑运算符进行组合。例如：（x>0）&（x<=3），其中 & 为逻辑运算符“与”。关系运算符与逻辑运算符分别参见表 1-2-2 和表 1-2-3。

表1-2-2　关系运算符

运算符	名　称	表达式	语　义
==	等于	x==y	x 与 y 相等时表达式值为 true，否则为 false
!=	不等于	x!=y	x 与 y 不相等时表达式值为 true，否则为 false
>	大于	x > y	x 大于 y 时表达式值为 true，否则为 false
<	小于	x < y	x 小于 y 时表达式值为 true，否则为 false
> =	大于等于	x > =y	x 大于等于 y 时表达式值为 true，否则为 false
< =	小于等于	x < =y	x 小于等于 y 时表达式值为 true，否则为 false

表1-2-3　逻辑运算符

运算符	名　称	表达式	语　义
&	与	（x > y）&（z!=0）	x 大于 y 且 z 不等于 0 时表达式值为 true，否则为 false
|	或	（x > y）|（z!=0）	x 大于 y 和 z 不等于 0 中至少有一个为真时，表达式值为 true，否则为 false

例如：如果 P1<P2，输出“n 千米时，乘甲车省钱”，用条件指令表示为：

```
if（P1<P2）输出（" n 千米时，乘甲车省钱 "）;
```

◆循环指令

在解决问题的过程中，有时需要重复同样的步骤，这就可以用循环指令来实现。

一般格式：

```
while（条件表达式）
{
    循环体；
    改变条件表达式的值；
}
```

语义：先判断“条件表达式”是否为真。若不为真（即为假），则不执行“循环体”，结束循环指令；若为真，则执行“循环体”一次，然后再判断“条件表达式”，若为真，则再次执行“循环体”……直到判断“条件表达式”的结果为假，结束循环指令。

设计循环指令时，循环体中必须包含能使“条件表达式”的值发生改变的指令，否则，就会导致程序出现死循环。

在学习了上述 4 种伪代码指令后，我们用伪代码来描述出租车计费问题的算法如下：

```
输入：甲、乙两车的起步价及其每千米的乘车费用和欲行驶的里程数
输出：相同里程数时，乘坐哪种车省钱
指令：P1←甲车的起步价；
      P2←乙车的起步价；
      X1←甲车起步价以后每千米的乘车费用；
      X2←乙车起步价以后每千米的乘车费用；
      n←欲行驶的里程数；
      if（n>=3）
      {
          P1←P1+X1*(n-3+1)；
          P2←P2+X2*(n-3+1)；
      }
      if（P1<P2）输出（n,"千米时，乘甲车省钱"）；
      else
      {
          if（P1>P2）输出（n,"千米时，乘乙车省钱"）；
          else 输出（n,"千米时，两车费用相同"）；
      }
```

程序设计的三种基本结构

在《信息技术基础》一书中，我们已经学习了程序设计的三种基本结构，即顺序结构、分支结构和循环结构。这三种结构虽然比较简单，但是可以用来描述适用于计算机求解的各种算法。

三种结构的指令执行顺序有所不同。顺序结构就是按照指令出现的先后次序依次执行。条件指令是分支结构，它判断所给的条件是否成立，从而选择某一条路径的指令继续执行。循环指令是循环结构，它首先判断条件是否成立，如果不成立则直接跳出循环体，执行循环体外的第一条可执行指令；如果成立则执行循环体内的指令，然后再次判断条件是否成立，如果成立则再次执行循环体内的指令，如果不成立则跳出循环体。

（3）流程图

流程图是用图形表示算法的一种常用工具，其优点是直观易读。流程图中常用的符号如图 1–2–1 所示。

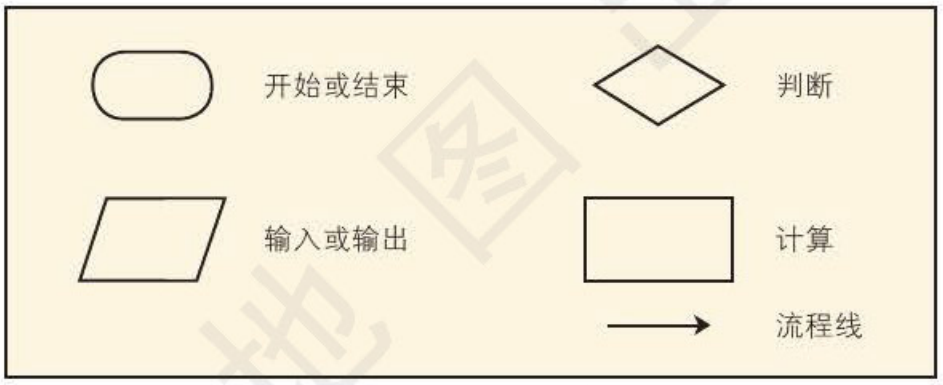


图1–2–1　流程图的常用符号

用流程图表示的三种基本结构见图 1–2–2。

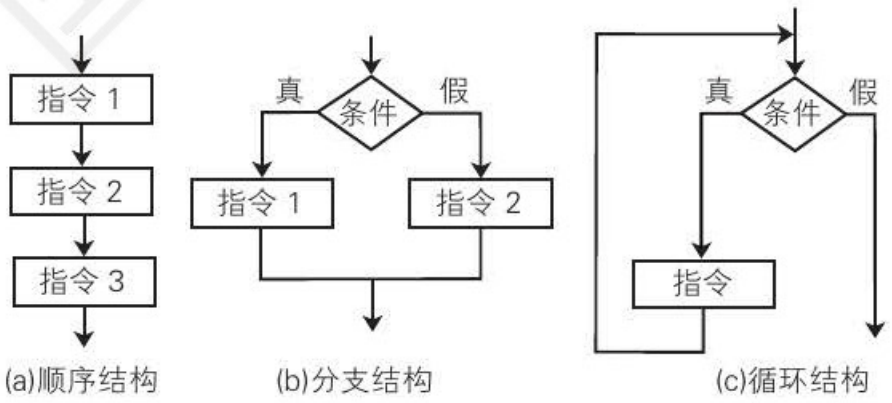


图1–2–2　用流程图表示的三种基本结构

例如：在出租车计费问题中，根据里程数计算乘车费用的步骤可以用图 1-2-3 表示出来，其中包含了顺序结构和分支结构。

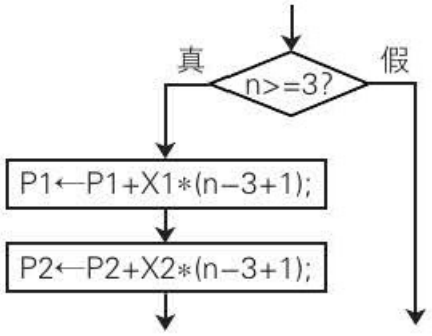


图1-2-3 出租车计费问题部分流程图

用流程图来表示出租车计费问题的完整算法。

用文档处理软件画流程图时，通常可在软件的绘图工具栏中找到常用的流程图符号，如图 1-2-4 所示。

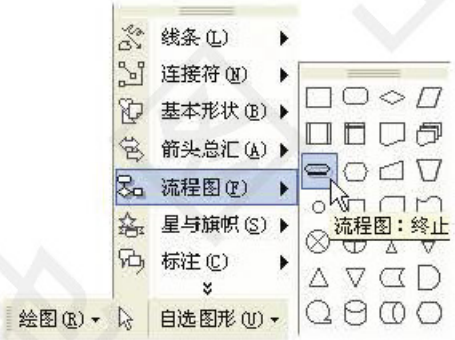


图1-2-4 绘图工具栏的流程图符号

（4）结构化流程图

流程图在表示算法上有一些缺点，如容易有三种基本结构之外的情况。美国学者南西（Nassi）和斯奈德曼（Shneiderman）两人提出了一种取代流程图描述算法的方法，这就是框图，又称 N-S 图。用框图表示的三种基本结构如图 1-2-5 所示。

(a)顺序结构　(b)分支结构　(c)循环结构

图1-2-5 用框图表示的三种基本结构

图 1-2-5（a）表示顺序执行指令 1、指令 2、指令 3。图 1-2-5（b）表示当条件成立时执行指令 1，当条件不成立时执行指令 2。图 1-2-5（c）表示当条件成立时执行图中的指令，之后再判断条件是否成立，只要条件成立则重复执行图中的指令；当条件不成立时，执行后续的指令。

使用框图表示算法要优于流程图，特别是用于描述流程比较复杂的算法，因为它可以保证只含有三种基本结构。但是，当修改算法需要重画框图时，工作量较大。

出租车计费问题的算法用框图表示如图 1-2-6 所示。

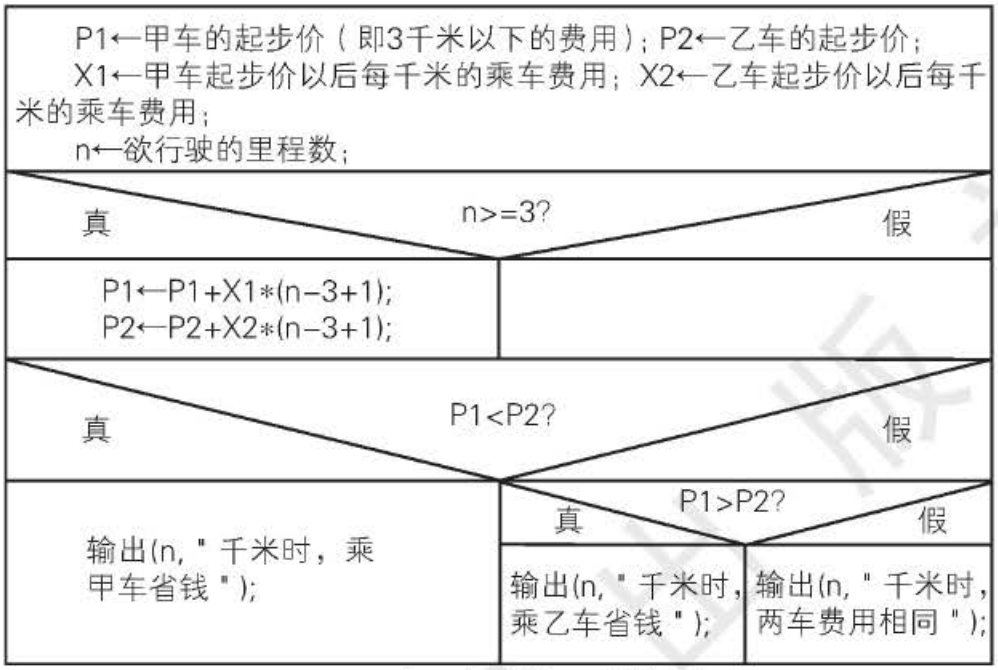


图1-2-6 出租车计费问题的算法框图

1．设计算法，计算全班同学的平均身高，并用学过的任一种方法进行算法描述。

2．因特网上有许多绘制流程图和框图的相关软件，请同学们用搜索引擎找一找，并下载下来试用。

3．如果想学习更多的关于算法的知识，可以上网找一找，这对于学习后面的内容是大有裨益的。

第三节 算法的程序实现

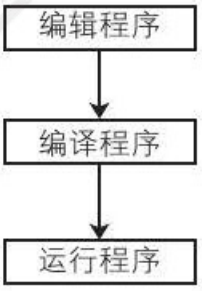
在经历了发现问题、分析问题、设计算法后，我们学会了如何对算法进行描述，但是算法并不能直接在计算机上运行，因此，我们还需要使用一种计算机能理解的程序设计语言将算法表达（翻译）成程序。本节我们一起来学习如何编写程序实现问题的求解。

在我们刚刚走进编程的大门时，会遇到选择什么样的编程语言和编程环境的问题。其实选择什么并不重要，关键在于掌握算法设计的方法。有了算法，只要熟悉一种编程语言就可以很快迁移到其他语言环境中去。

一 编写程序

由于 Java 是目前已被广泛使用的、可支持网络应用的程序设计语言，所以我们采用 Java 语言来实现算法的程序化。

编写与运行一个完整的 Java 程序的流程是：



1. 程序的编辑

Java 源程序可以使用任何纯文本编辑工具编写，例如 Windows 自带的“记事本”。

现在，我们使用“记事本”来编写求解出租车问题的 Java 程序，输入出租车计费问题的程序代码，如图 1-3-1 所示。保存时，文件名必须与程序代码中“public class”后面的名字完全相同，文件扩展名为“.java”。

“public class”的意思是定义公共的类，如：public class TaxiBill

其中 TaxiBill 是公共类的名字，所以源程序保存时的文件名必须是“TaxiBill.java”。

TaxiBill.java - 记事本

文件(F)　编辑(E)　格式(O)　查看(V)　帮助(H)

```
//程序名称：TaxiBill.java
//程序功能：根据里程数计算乘坐哪种出租车省钱
import javax.swing.JOptionPane;
public class TaxiBill                          //TaxiBill为公共类的名字，与程序文件名一致
{
    public static void main(String args[])    //程序执行的起点
    {
        double P1=10;                          //P1为乘坐甲车的费用，P1←甲车的起步价
        double P2=8;
        double X1=2;                           //X1←甲车3千米以上（含3千米）每千米的费用
        double X2=2.2;

        String s=JOptionPane.showInputDialog("请输入行驶的里程数");
        int n=Integer.parseInt(s);             //n←欲行驶的里程数

        if (n>=3)
        {
          P1=P1+X1*(n-3+1);
          P2=P2+X2*(n-3+1);
        }

       //以下是输出结果
       if (P1<P2) JOptionPane.showMessageDialog(null,n+"千米时，乘甲车省钱","出租车的计费问题",
JOptionPane.INFORMATION_MESSAGE);
       else
       {
          if (P1>P2) JOptionPane.showMessageDialog(null,n+"千米时，乘乙车省钱","出租车的计费问题",
JOptionPane.INFORMATION_MESSAGE);
          else JOptionPane.showMessageDialog(null,n+"千米时，两车费用相同","出租车的计费问题",
JOptionPane.INFORMATION_MESSAGE);
       }

    }
}
```

图1-3-1　使用“记事本”编辑源程序

2. 程序的编译

Java 的源程序必须翻译成 Java 系统能识别的专用文件才能执行，这一过程叫“程序的编译”，执行这一过程的程序叫“编译程序”。Java 系统中编译程序的名字叫“javac.exe”。编译后生成的 Java 专用文件由字节码组成，一般人无法识别，这种文件在 Java 系统中叫类（class）文件，扩展名为“.class”。

假设 Java 源程序“TaxiBill.java”存放在“D:\jdk”下，用下列步骤进行编译（如图 1-3-2 所示）：

①进入系统的“命令提示符”界面；

②进入“D:\jdk”目录；

③输入“javac TaxiBill.java”后回车。

如果编译程序未返回任何提示信息，就说明程序运行完成，结果是产生一个“TaxiBill.class”文件，存储在与源文件“TaxiBill.java”相同的目录中。

图1-3-2　编译步骤示例

编译程序 javac.exe 从哪里获取

Java 系统一般由四部分组成：Java 环境、Java 语言、Java API 应用程序接口和 Java 类库。这四部分内容包含在一个 Java 开发工具集（Java Development Kit，简称 JDK）中。Java 系统软件商业版本的正式名称叫“Java 2 Software Development Kit, Standard Edition”，简称“J2SE SDK”。最新版本的 SDK 可以从因特网上免费获取。

本书的配套光盘中“zy\tools”目录下提供了“j2sdk-1_4_2.exe”，运行该文件则自动安装 Java 语言的开发环境。如果将 Java 安装在“D:\jdk”目录中，Java 系统的可执行文件就位于“D:\jdk\bin”目录下，编译程序“javac.exe”文件即存于此。

系统安装好后，Java 系统的程序都存放在 JDK 安装目录的子目录“\bin”下。使用 Java 程序之前，还应该设置操作系统的环境变量。

①把系统路径（path）指向“\jdk\bin”；

②设置 classpath 指向 Java 的 class 文件所在目录。

（提示:Java 的执行文件存放于“\jdk\bin”目录,类文件存放于“\jdk\lib\tools.jar”中）

例如，将这两条指令加入到 C 盘根目录下的“autoexec.bat”文件中，如图 1-3-3 所示。

图1-3-3 修改“autoexec.bat”的环境变量

3. 程序的运行

Java 程序编译后生成的新文件“TaxiBill.class”是类（class）文件。类文件不能在计算机上独立运行，必须使用解释程序加以解释才可以执行和输出结果。Java 系统“D:\jdk\bin”目录下的可执行文件“java.exe”就是解释程序，它可以解释我们生成的“TaxiBill.class”程序，使程序运行。

程序正确编译后，直接输入“java 类文件名”就可以运行程序，需要注意的是此处的类文件名不能带扩展名，否则会产生错误。正确的程序运行步骤如图 1-3-4 所示。

图1-3-4 程序运行步骤示例

“出租车计费问题”程序的运行结果如图 1-3-5 和图 1-3-6 所示。

图1-3-5 程序运行结果的输入对话框

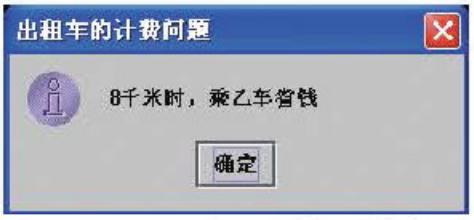


图1-3-6 程序运行结果的输出

在配套光盘中找到出租车计费问题的Java源程序，编译后运行，观察结果。了解算法的程序实现过程。

Java 程序有两种：一种是独立的应用程序，叫做 Application 程序；另外一种是依附在网页上的程序，叫 Applet 小程序。这两种程序的代码不尽相同，其运行方式也不相同。

应用程序的运行是在命令提示符窗口下运行解释程序“java.exe”来执行的。

如图 1-3-7 所示，Applet 小程序的运行一是通过在命令提示符窗口运行“appletviewer.exe”程序来执行，它调用 Applet 小程序所依附的网页，执行时产生一个 Applet 窗口显示运行结果；二是通过浏览器来运行。

（a）命令提示符窗口及Applet 窗口运行结果

（b）浏览器运行

图1-3-7 Applet小程序的两种运行方式

并非所有的浏览器都可以执行 Applet 小程序，如果你的浏览器不能执行，请到因特网上搜索并下载相应的嵌入程序（Plug-In）。本书配套光盘中“zy\tools”目录中有一个嵌入程序“j2re-1_4_2.exe”，请大家试着安装一下。

认识Java程序的基本结构，尝试编写简单的Java程序。

图 1-3-8 是一个 Java 应用程序的代码，图 1-3-9（a）和 1-3-9（b）分别是一个 Applet 小程序及网页中嵌入这段小程序的 Applet 代码。

①分别输入这两种程序的代码，编译并运行程序，说明运行结果。

②分别从程序代码和运行结果来比较这两种程序的异同。

③试对程序代码中加下划线的字符串和数值进行修改，看看运行结果会发生什么变化。你能说出加下划线的这条命令起什么作用吗？

图1-3-8 Java应用程序代码

（a）Applet小程序的代码

（b）网页中嵌入的Applet代码

图1-3-9 Applet小程序

Java的编辑软件

用“记事本”编辑Java程序，在“命令提示符”窗口编译并运行Java程序会很麻烦。有没有其他比较方便的工具软件呢？其实，因特网上有许多免费或供试用的小软件，例如，有一个叫“TextPad”的工具软件就能较方便地编辑、调试、调用JDK编译和运行Java程序。它的主界面如图1-3-10所示。

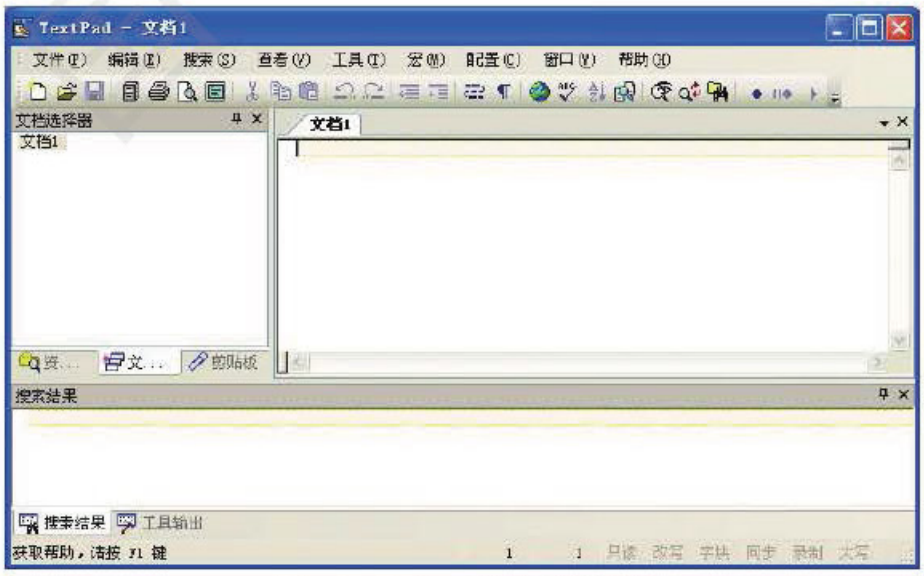


图1-3-10 TextPad的主界面

TextPad 存放在本书配套光盘的“zy\tools”目录下。它的安装非常简单，将软件包解压后，直接运行“TextPad.exe”即可。

使用这个软件，一般先要设置适合 Java 程序编辑、运行的快捷方式。具体操作是设置菜单【Configure】中的【Preferences】，选择“Tools”条目，在右侧窗口中添加“Java SDK Commands”，于是“Compile Java”“Run Java Application”和“Run Java Applet”三项快捷功能就被添加上了。

当你打开或创建一个新的 Java 源程序时，直接按 <Ctrl+1> 键就可以对其进行编译，按 <Ctrl+2> 键可运行编译好的类文件（Java 的应用程序），按 <Ctrl+3> 键运行一个 Java Applet 小程序时会带来意想不到的效果，即不用编写 html 源代码就可以显示 Java Applet 程序的运行结果。

需要明确的一点是，TextPad 必须在预先安装了 JDK 集成工具集之后才有这么强的功能。因为它的上述三个功能不过是调用 JDK 集成工具集中的“javac.exe”“java.exe”以及“appletviewer.exe”而已。

类似的工具软件还有很多，比如 JCreator、JPad Pro、EditPlus、UltraEdit 等。

虽然 JDK 开发工具可以用来编辑、编译、运行 Java 程序，但要编制、调试较复杂的应用系统就不方便了。为更加快速、高效地开发 Java 程序，可以使用可视化集成开发环境软件（简称 IDE），如 Borland JBuilder，IBM Visual Age for Java，WebGain Visual Café，Oracle Internet Developer。

这些复杂的 Java 集成开发环境虽然功能强大，但对初学者来说不如直接使用 JDK 和文本编辑器好，这样可以把精力集中在 Java 语言本身，因为最终实现算法才是我们的目的。

二　调试程序

编译程序时，很难一次就获得成功，通常会出现这样或那样的错误，可能是语法错误，也可能是逻辑错误。在编程过程中，要善于把遇到的问题进行归纳总结，以避免再犯同样的错误。

1. 语法错误

语法错误是由于指令没有按照程序设计语言的语法规则编写所致。例如，当打印一个消息时，如果程序的打印命令不正确，就会产生语法错误。Java 语言对字符的大小写是非常敏感的，一些中文标点符号也极易混杂在程序代码中造成错误。

2. 逻辑错误

这种类型的错误通常在程序运行时发生，虽然语法正确但并不能产生所需要的结果。这类错误可能是由数据类型错误造成的，也可能是运算符号搞错了，还可能是算法不完整、不正确造成的。逻辑错误比语法错误通常更难察觉。

为了避免这类错误，必须对程序进行测试。测试时要输入测试数据来看看程序是否能够产生正确的结果。如果程序没有产生正确结果，必须查找程序中的错误，修改错误，然后再进行测试。调试程序的过程可能需要相当长的时间，但这是程序设计中的关键，不可马虎。一般情况下测试一组值是不够的，至少应当测试程序的各个分支。

知识拓展 程序编译与运行时常见的错误信息

1. javac:Command not found

表示找不到“javac.exe”程序，可能是由于路径未正确设置。

2. can't find class Taxibill

表示不能找到类文件“Taxibill.class”，可能是由于在命令行中所指定的类名“Taxibill”的拼写与“TaxiBill.class”文件名的拼写不同所致。因为 Java 是一种对字母大小写敏感的语言，因此要求拼写严格一致。

3. variable x might not have been initialized

通常表示变量 x 未初始化。

4. ';'expected 或 '}'expected

表示缺少“;”或“}”。

5. can't resolve symbol

通常表示变量未声明。

6. incompatible types

表示数据类型不匹配。

三 程序文档

程序文档是解释程序的工作过程及使用方法的文档，它可以帮助我们理解当前程序的工作过程，能为我们修改程序或者使用程序提供便利。一个计算机程序不可避免地要进行修改，如果没有程序文档，理解和修改一个很长的程序是比较困难的。特别是当修改别人编写的程序，或者重读、修改很久以前自己编写的程序时，困难就可想而知了。

程序文档一般有两种形式：插入到程序代码中的注释和专门制作的文档。

1. 程序代码中的注释

注释是插入到计算机程序代码行中的说明，可以解释程序功能，也可以解释一些难懂的程序段。计算机编译执行程序时会略过这些注释。我们在编写程序时要养成加注释的良好习惯，有了注释，不论什么时候读起程序来都比较容易理解。当需要改写程序时，首先需要读懂原来的程序，还要注意为修订的代码段修改注释。

什么地方该加注释并没有固定的法则，但开发一个软件，项目小组一般会制定一些规定，要求大家在程序中使用一致的注释。如在程序中注明程序名称、功能、作者以及开发时间等内容。

Java 程序中的注释一般有两种格式：

```
// 注释内容——表示从“//”开始到本行末的所有字符均为注释；
/* 注释内容 */——表示从“/*”开始到“*/”之间的所有字符均为注释。
```

例如，图 1-3-1 中所示的出租车计费问题源程序中就加入了程序名称和功能等注释。

2. 专用的程序文档

专门制作的程序文档不属于程序，它只包含一些关于程序的信息。文档既可以是书面的，也可以是电子表格式的。一般包含两部分：程序手册和用户参考手册。

程序手册中包含所有对程序员有用的信息，如问题描述和算法等，有时还提供程序的源代码。程序手册是进行软件开发和维护的依据，通常由程序员对程序代码进行编排、整理而成。

用户参考手册是对程序功能的说明，可以帮助用户尽快学会使用该软件。包括如何安装、启动和运行程序，如何使用程序的图表和菜单，以及对故障的分析与排除等等。用户参考手册的电子版本通常能给用户提供在线式帮助。用户参考手册现在一般由专业人员编写，专门解释一些专业术语和程序，把复杂的概念加以简化，供非技术人员阅读。

关于 Java SDK 的程序文档，请到因特网上查找相应资料，在那里你可以学习到更多关于 Java 的知识。

Java 技术的应用

随着互联网的发展，技术的更新迭代，Java 作为目前流行的计算机程序设计语言之一，在许多领域中都有应用。

1. Android 应用

许多 Android 应用（如一些手机游戏）都是基于 Java 程序开发的。虽然 Android 运用了不同的 JVM（Java 虚拟机）以及不同的封装方式，但是代码还是用 Java 语言所编写。

2. 应用于金融业的服务器程序

在金融服务业，Java 的应用非常广泛，很多第三方交易系统、银行、金融机构都选择用 Java 语言来编写前台和后台的电子交易系统、结算和确认系统、数据处理项目以及其他项目等。

3. 网站

Java 在电子商务领域以及网站开发领域占据了一定的席位，开发人员可以运用许多不同的框架来创建 Web 项目，如医疗、保险、教育等许多网站都是以 Java 为基础来开发的。

4. 嵌入式领域

Java 在嵌入式领域发展空间很大，包括无线手持设备、智能卡、通信终端、医疗设备、信息家电（如数字电视、机顶盒、电冰箱）、汽车电子设备等都是近年来热门的 Java 应用领域。

此外，Java 还可以显示图形界面、进行数学运算和网络操作以及对数据库和文件进行操作，甚至可以进行自然语言处理等。

1．将前面已经在“命令提示符”窗口中运行过的“Hello.java”和“HelloApplet.java”程序在 TextPad 环境下运行，你会发现在两种环境下运行的程序在语法和格式方面有些不同，请你指出这些不同。你还能发现其他不同之处吗？

2．从因特网上下载一个简单的 Applet 小程序（源代码），先在“命令提示符”窗口下编译运行，然后在 TextPad 环境下运行，再直接利用浏览器运行。对你能读得懂的代码部分作适当的修改，然后重新编译、调试程序，并查看运行结果。

第四节 程序设计语言简介

本节我们将了解计算机程序设计语言的特点，学会根据需要选择合适的编程语言。

在使用计算机的过程中，我们会接触到大量的计算机软件，如文字编辑软件、图形绘制软件、网页浏览软件、邮件收发软件，等等。这些计算机软件都是使用某种程序设计语言开发出来的。下面，让我们来了解一下程序设计语言吧！

一 程序设计语言的发展

自电子计算机问世以来，程序设计语言的发展大致经历了机器语言、汇编语言、高级程序设计语言等几个阶段。

1. 机器语言

20 世纪 40 年代，最早的电子计算机只能识别机器语言，这种机器语言是 CPU 可以识别的一组由二进制数 0 和 1 序列构成的指令码。

用机器语言编程序，就是直接使用 CPU 指令系统中的指令，组成一个指令序列。这种程序可以被机器直接理解和执行，但是它们不直观，难记、难认、难理解，同时程序的编写效率很低，质量难保证。机器语言是第一代计算机语言。

2. 汇编语言

20 世纪 50 年代中期，为了解决使用机器语言编程的困难，人们进行了一种有益的改进，开始用一些“助记符号”来代替二进制数编程。比如，用“ADD”代表加法，“MOV”代表数据传递，等等。这样一来，人们很容易读懂并理解程序的意义，对程序进行纠错及维护都变得方便了。这种程序设计语言就称为汇编语言，即第二代计算机语言。

然而计算机不能识别汇编语言的符号，因此需要一个专门的程序来将这些符号翻译成机器语言，这种翻译程序被称为汇编程序。

汇编语言同机器语言一样是十分依赖于机器硬件的，移植性不好，但执行效率非常高，针对计算机特定硬件编制的汇编语言程序，能很好地发挥计算机硬件的功能和特长。用汇编语言编写的程序精炼且质量高，所以汇编语言至今仍是一些专业程序员常用的软件开发工具。

3. 高级语言

随着计算机的不断发展，人们意识到，应该设计一种这样的语言：它接近数学语言或人的自然语言，同时又不依赖于计算机硬件，编写的程序能在通用计算机上运行。1954 年，第一个完全脱离机器硬件的高级语言——FORTRAN 问世了。此后几十年，共有几百种高级语言出现，其中具有重要意义的就有几十种，影响较大、使用较普遍的程序语言有 FORTRAN，COBOL，BASIC，Lisp，Pascal，C，Prolog，C++，Java 等等。

高级语言的发展经历了从非结构化到结构化，从面向过程到非过程化的过程。相应地，软件的开发也由最初的个体手工作坊式的封闭式生产，发展成为产业化、流水线式的工业化生产。

20世纪60年代中后期，软件越来越多，规模越来越大，而软件的生产基本上是各自为战，缺乏科学规范的系统规划与测试、评估标准，结果是大批耗费巨资建立起来的软件系统由于含有错误而无法使用，甚至带来巨大损失。软件给人的感觉是越来越不可靠，出现了“软件危机”，震惊了计算机界。人们认识到：大型程序的编制不同于写小程序，它应该是一项新的技术，应该像处理工程一样处理软件研制的全过程。程序的设计应易于保证正确性，也便于验证正确性。

1969年，有人提出了结构化程序设计思想。1970年，第一个结构化程序设计语言——Pascal出现，它标志着结构化程序设计时期的开始。从20世纪80年代初开始，在软件设计思想上又发生了一次革命，其成果就是提出了面向对象的程序设计。其方法就是将软件集成化，如同硬件的集成电路一样，生产一些通用的、封装紧密的功能模块，它与具体应用无关，但能相互组合，完成具体的应用功能，同时又能重复使用。对使用者来说，只需要关心它的接口（输入和输出）及能实现的功能就可以了，至于如何实现的，那是它内部的事，使用者完全不用关心。C++、Java就是面向对象的程序设计的典型代表。

高级语言的下一个发展目标是面向应用，也就是说：只需要告诉程序你想干什么，程序就能自动生成算法，自动进行处理，这就是非过程化的程序语言。

4. 第四代语言

第四代语言（Fourth—Generation Language，4GL）是按计算机科学理论指导设计出来的结构化语言，如ADA，MODULA—2，SMALLTALK—80等。4GL具有简单易学，用户界面良好，非过程化程度高，面向问题等特点，只需告知程序“做什么”，而不必告知程序“怎么做”，程序就能自动生成算法，自动进行处理。

二　选择合适的编程语言

在设计程序之前，从系统开发的角度考虑选用哪种语言来编程是很重要的，一种合适的语言能减少编程的工作量，缩短设计程序的时间，并且可增强程序的可读性和可维护性。

1. 选择程序设计语言的原则

通常情况下，一项任务可以由多种编程语言实现。当选择一种编程语言时一般要综合考虑以下几方面的情况：

第一，语言的集成环境和交互功能；

第二，语言的结构机制和数据管理能力；

第三，可移植性和开发人员的熟练程度；

第四，是否有较多的使用者；

第五，用户有无特殊的要求；等等。

2. 常用的高级编程语言

（1）BASIC与Visual Basic

BASIC是为初级编程者设计的。自1964年问世以来，BASIC已经出现了几种流行版本，包括GW-BASIC和QBASIC。由于BASIC容易使用并且适合各种计算机系统，所以它曾经是最流行的程序设计语言之一。早期的BASIC是一种面向过程的高级语言，它的大多数版本都是解释执行的语言，但也有一些是编译执行的语言。近年来出现的Visual Basic是Windows下的可视化编程语言环境，易于开发使用，支持面向对象的程序设计。

（2）Pascal与Delphi

Pascal 开发于 1970 年，主要用于帮助学生学习计算机编程。Pascal 是编译执行的面向过程的高级语言。它开创了结构化程序设计的先河，但 Pascal 很少用于专业编程和商用软件的开发。近年来出现的 Delphi 以面向对象的 Object Pascal 为基础，应用比较广泛。

（3）C与C++

C 语言是编译执行的面向过程的高级语言并带有低级语言的接口，这种特性给程序员带来很大的灵活性，有经验的程序员利用这种灵活性可以提高程序的运行速度和工作效率，但这种特性也使得 C 程序难以理解、调试和维护。

C++ 是支持面向对象的 C 语言。C++ 的面向对象特性大大提高了程序员的开发效率。

（4）C#

C#（可读做“C sharp”）是一种面向对象运行于 .NET Framework 之上的高级程序设计语言。C# 继承自 C 和 C++，综合了 Visual Basic 简单的可视化操作和 C++ 的高运行效率，具备便捷的面向组件编程的支持，是 .NET 开发的首选语言。

（5）Lisp和Prolog

Lisp 和 Prolog 是说明性高级语言，一般用于开发专家系统。它们分别开发于 1960 年和 1971 年。说明性的高级语言只需具体说明问题的规则并定义一些条件，语言自身内置的方法就能把这些规则解释为一些解决问题的步骤，这种把编程重心转移到描述问题和它的规则上的处理方法，更适合思维概念清晰但数学概念复杂的编程工作。例如在人工智能技术领域就广泛使用 Prolog 语言。

（6）Java

Java 是一种完全面向对象的高级语言，而且是因特网应用的主要开发语言之一。它的运行与操作系统平台无关，这就意味着 Java 应用程序不但能在 Windows 系统上运行而且可在 MacOS 和 UNIX 系统上运行。Java 是当今最具活力、发展最快、最具有网络应用优势的编程语言。Java 的应用已深入到各个领域，如：Web 服务器、关系数据库、大型计算机、移动电话、空间技术、家电和各种智能卡，等等。Java 的 Applet 小程序尤其适于生成网页上栩栩如生的图画。但 Java 的运行速度还有待提高和改善。

本书的程序设计示例选用的就是 Java 语言。

下载感兴趣的程序。

到因特网或教科书的配套光盘中检索一下 Java 在网页开发方面的特效，如果有比较感兴趣的程序，请下载下来，存放到你的电子学习档案袋中，并进行有效归类，以备后面的学习使用。

选择一种你感兴趣的程序设计语言，到因特网上浏览或下载相关学习资料，并试着回答以下问题：

① 它是面向对象的语言，还是面向过程的语言，或是其他语言？

② 这种语言的开发背景是什么？

③ 谁是这种语言的开发者？

④ 这种语言的发展趋势如何？

第二单元·程序设计基础

只要涉及编程的问题，就不可避免地要选择一种程序设计语言进行算法的实现。良好的开发环境只是编写一个好程序的必要条件，而不是充分条件，因此我们不需要过多考虑该选择哪种编程语言或什么样的编程环境，关键是要一开始就注重培养合理的逻辑思维和科学的创造理念。

首先我们一起来学习程序设计语言的基本知识，在掌握了模块化程序设计的基本思想后，从剖析他人编写的程序代码开始，尝试通过修改代码来解决自己的实际问题。这也是代码重用的思想。重用不是偷懒，而是向前人学习，因为人类总是在继承前人成果的基础上不断加以利用、改进或创新后才进步的。面向对象程序设计人员有一句口头禅是“请不要再发明相同的车轮子了”，你们对这句话会有怎样的感想呢?

第一节 数据及其运算

常量、变量、数据类型、运算符和表达式等基本知识的学习是掌握任何一种程序设计语言的前提。我们将以 Java 程序设计语言为基础，学习并灵活运用这些知识。

在学习常量、变量、数据类型、运算符和表达式等基本知识之前，我们先来了解 Java 程序的结构。

一 程序结构

程序结构通常指一段完整的程序代码是由哪些内容组成的。Java 编程语言同其他常见的编程语言一样，可用来创建应用程序。下面的程序代码就是一个简单的 Java 应用程序。

程序：Sum1.java

功能：计算两个数的和

```
// 注释行。这是一个简单的能计算两个数之和的 Java 程序
public class Sum1                                    //Sum1 为程序文件名
{
    public static void main(String args[ ])        // 程序执行的起点
    {
       int a=3;                  // 声明 a 为整型变量，并把 3 赋给 a
       int b=5;                  // 声明 b 为整型变量，并把 5 赋给 b
       int c;                    // 声明 c 为整型变量
       c=a+b;                    // 计算 a 与 b 的和，并赋给 c
       System.out.println(" 计算两个数的和 :"); // 输出双引号中的字符
      System.out.println(a+"+"+b+"="+c); // 输出运算结果，双引号外的加号表示接连输出
    }
}
```

从这段程序代码中，我们可以了解 Java 程序的结构。

Java 程序的结构包括三个部分：

◆ Java 程序的注释（Comment）语句

单行注释：以双斜线“//”开始到本行末尾。

多行注释：以“/*”开始，以“*/”结束。

◆主类

public class Sum1

```
{
        ……
}
```

这部分称为程序的主类。Java 的程序是由许多类（类通过关键字 class 来声明）组成的，类中包含许多数据及处理这些数据的方法。public 是一个公共类，它表示允许其他任何类访问。它以左花括号“{”开始，以右花括号“}”结尾。主类名 Sum1 就是保存文件的 Java 源程序的文件名，主类名与文件名必须完全一致。

◆主方法

```
public static void main(String args[ ])
{
……
}
```

这部分称为 Java 程序 Sum1 类的 main( ) 方法（通常称为主方法），是程序执行的起点。同样，它以左花括号“{”开始，以右花括号“}”结尾。从这部分开始的代码按照程序控制的顺序执行。

main( ) 方法通常由若干语句构成。在 Java 编程语言中，语句是一行由分号“;”终止的代码。一个块（block）或一个复合语句是以花括号“{ }”为边界的语句集合。

其中，“String args[ ]”叫做 main( ) 方法的参数。

注：Java 程序中允许任意多的空白。

书写程序时，通常采用缩进格式，这样可以使程序结构更加清晰。

二　数据类型

Java 提供了四种基本数据类型：整数型、浮点型、字符型、逻辑型，见表 2-1-1 所示。

表2-1-1　Java的四种基本数据类型

类　型	类型名称	所占空间	缺省值	数据范围
整数型	byte	1 个字节	(byte)0	−128 ~ 127
	short	2 个字节	(short)0	−32768 ~ 32767
	int	4 个字节	0	−2147483648 ~ 2147483647
	long	8 个字节	0L	−9223372036854775808L ~ 9223372036854775807L
浮点型	float	4 个字节	0.0F	绝对值在 1.40239846×10^{-45} ~ 3.40282347×10^{38} 之间
	double	8 个字节	0.0	绝对值在 $4.94065645841246544\times10^{-324}$ ~ $1.79769313486231570\times10^{308}$ 之间
字符型	char	2 个字节	'\u0000' 或 null	'\u0000' ~ '\uFFFF'，前缀 \u 表示一个 Unicode 字符编码
逻辑型	boolean	1 个比特	false	true 或 false

所有高级语言在进行程序设计时，都涉及常量、变量和数据类型（Data Type）等概念。每一个数据都属于一个特定的数据类型。

例如语句：`int a=3;`

表示将 a 声明为整型变量（int），并赋初值 3。

1. 常量

常量（Constant）的值在程序运行过程中不能改变。

◆整型常量：即整常数。

如 123，-123，0。默认为 int 类型。

例如语句 :long a=-257L ;（数字后跟字母 L 或 l 表示 long 型）。

再如：int a=3;。声明 a 为整型变量，并赋初值 3（整常数）。

int b=3.0;。为错误写法（3.0 不是整型常量）。

◆浮点型常量：或称浮点数（即实数），有两种形式。

十进制形式，如 0.168，.123，-23.45 等。

指数形式，如 123e2 或 123E2 都表示 $123*10^2$。注意字母 e（或 E）之前必须有数字且 e 后面的指数必须为整数。

浮点数用科学记数法表示。

例如：-6.12E-18 表示 -6.12×10^{-18}。

-18　指数（为整数）

E　指数符号（E 表示乘以 10 的幂）

-6.12　尾数（为实数）

例如：float a=0.1f（或 float a=0.1F）表示单精度浮点数。double b=5.6（或 double b=5.6D、double b=5.6d）表示双精度浮点数，默认为 double 型。

◆字符常量

用单引号括起来的单个字符，如 'a'，'A'，'2' 等。注意 'a' 和 'A' 是两个不同的字符。

例如：char a='b';

对于无法用键盘输入的字符，则用转义符来表示，如“\r”表示回车，“\n”表示换行。

例如：char b='\n' ;

◆逻辑常量

逻辑常量只有两个值，即 true 和 false，代表两种状态，即真和假、是和非，true 和 false 这两个值不能加单引号且只能小写。

例如：boolean a=true ;

◆字符串

用双引号括起来的一串字符，称为字符串（Character String）。

如果一个字符串不包含任何字符，则称为空串。在 Java 中，使用 String 来声明字符串。String 不属于基本数据类型，而是一个类（Class），它被用来表示字符序列。例如：

```
String s= "Good Morning!" ;
```

此外，还可以在程序中定义一些常量。例如：

```
final double PI=3.14159265;
```

其中修饰符 final 表示声明 PI 为常量（圆周率）。

2. 变量

变量（Variable）是指在程序运行过程中，其值可以改变的量。变量是内存中存放数据的存储单元，存放的数据叫变量的值，用变量名来表示。

变量名必须以字母开头，后跟字母或数字。这里的字母不仅仅是“a”到“z”和“A”到“Z”，还包括“$”“_”等，但不能是“%”“?”“#”“,”。如：my_name，$myboy，myclass 等都是合法的变量名。Java 严格区分大小写。

为了便于理解，变量名最好具有一定含义，但不能和 Java 关键字 (Keyword) 同名。

变量在使用之前必须先声明再使用。声明的语法格式为：

```
类型 变量名 [ = 初值 ][, 变量名 [= 初值 ]…] ;
```

其中 [ ]：表示可选。

例如：

```
int  x;              // 声明一个整型变量 x
double  a;           // 声明一个双精度浮点型变量 a
boolean b;           // 声明一个布尔型变量 b
……
int a,b;             // 声明 a，b 为整型变量
int c=300;           // 声明 c 为一个整型并附初值 300
……
a=100;
b=200;
……
a=101;
……
```

从上述部分程序代码可以看出，声明 a，b，c 为整型变量，程序执行到 a=100 时，a 的值为 100，然后程序继续执行，执行到某处，a 的值会发生变化，即变为 101。

在需要解决的实际问题中，大量数据并不是孤立的，而是有联系的。在高级语言中，可以用数组（Array）处理这类问题。也就是说可以把有某种关系的、类型完全相同的数据按照某一顺序放在一起，这组数据组成一个数组。数组的成员叫做数组元素，数组元素也是变量，数组元素的顺序号称为下标。

例如，数组 a 中有 5 个元素，可表示为：

a[0]	a[1]	a[2]	a[3]	a[4]
12	8	1	9	64

其中，a[n] 中的 n 表示数组元素的下标（n=0，1，2，3，4），a[0] 值为 12，a[1] 值为 8，……

数组的元素还可以是数组。如果数组的每一个元素都是数组，那么这个数组就是二维数组。

变量的类型可以是基本数据，也可以是数组或类。变量的数据类型决定了变量所占内存空间的大小。

数组在使用前也必须要声明类型。

例如：

```
String c[ ];        // 声明一个字符串型数组 c
int d[ ][ ];        // 声明一个整型的二维数组 d
```

三 运算符

在 Java 语言中，利用运算符（Operator）操作数据。运算符就是进行某种运算的标识符号，它标明对操作数据所进行的运算。

运算符以一个或多个变量为基础，可生成一个新值。运算符根据操作数据个数的不同可以分为一元运算符、二元运算符、三元运算符；根据运算功能又可以分为算术运算符、连接运算符、关系运算符、逻辑运算符、赋值运算符和其他运算符。

下面介绍算术运算符、连接运算符、赋值运算符。

1. 算术运算符

算术运算符可以进行算术运算，常见的算术运算符见表 2-1-2。在使用中要注意，它们的写法与数学中的写法有不同之处。

表2-1-2 算术运算符

算术运算符	描 述	用 法	举 例
+	加	a+b	2+3 的结果为 5
-	减	a-b	2-3 的结果为 -1
*	乘	a*b	2*3 的结果为 6
/	除	b / a	3 / 2 的结果为 1, (double)3/2 的结果为 1.5
%	求余	b%a	3%2 的结果为 1
++	递加	a++，++a	a=2; a++；a 的值等于 3，++a；a 的值等于 3
--	递减	a--，--a	a=2; a--；a 的值等于 1，--a；a 的值等于 1
+	取正值	+a	a=2; +a；a 的值等于 2
-	取负值	-a	a=2; -a；a 的值等于 -2

2. 连接运算符

连接运算符“+”可以把两个或多个字符串连接起来，这种操作也叫连接运算。

例如："ABC"+"123" 的运算结果是 "ABC123"；

"123"+"456" 的运算结果是 "123456"。

3. 赋值运算符

（1）表达式

表达式（Expression）就是按语法规则由运算符将常量、变量或能实现某种功能的方法连接而成的有意义的式子。一个表达式无论简单或复杂，都要按规定的运算顺序进行运算，最后得到的结果称为表达式的值。一个表达式的所有字符必须写在同一行内。常量、变量或方法可看成表达式的特例。例如：

表2-1-3 代数式与表达式的对应关系

代数式	表达式
$\frac{a}{b+c}$	a / (b+c)
$\sqrt{2x}$	Math.sqrt（2*x）（此处使用了 Math 类的 sqrt（ ）方法）

如果两个或多个运算符出现在同一个表达式中，则按照优先原则（Precedence Rule）确定运算顺序。常用算术运算符的优先原则见表 2-1-4。

表2-1-4 常用算术运算符的优先原则

优先级	算术运算符
1	（）
2	一元运算符（正负号：+，－）
3	*，/，%
4	+，－

数值表达式只允许使用圆括号，当有几层圆括号时，内层优先。同一优先级的运算符，自左至右依次执行。

（2）赋值运算

赋值运算（Assignment）是将赋值运算符“=”右边表达式的值赋给赋值运算符左边的变量。例如：

```
a=3; // 赋值运算：将 3 赋值给变量 a
```

在 Java 程序中，任何变量都必须初始化后才能使用，初始化可以通过赋值运算来实现。例如：

```
String s ="Good Morning! ";      // 声明 s 为字符串，并进行初始化
int a[ ]= {1,2,3,4,5};           // 声明并初始化整型数组 a，a[0]=1，a[1]=2，……
```

为了简化程序，在赋值运算符“=”之前可以加上其他算术运算符构成扩展赋值运算符，见表 2-1-5。

表2-1-5 扩展赋值运算符

运算符	用　法	等效表达式	举　例
+=	a+=b	a=a+b	a=3; a+=2; 则 a 的值等于 5
−=	a−=b	a=a−b	a=3; a−=2; 则 a 的值等于 1
*=	a*=b	a=a*b	a=3; a*=2; 则 a 的值等于 6
/=	a/=b	a=a/b	a=5; a/=2; 则 a 的值等于 2
%=	a%=b	a=a%b	a=5; a%=2; 则 a 的值等于 1

知识拓展 Java语言中的类型自动转换

Java 是一种强类型语言，在进行一些操作运算时，先进行类型检查。如果类型不一致，就会按照某种规则自动转换，然后按照转换后的类型进行操作；若类型不一致且不能转换时，则会报错。

类型自动转换只能将低级数据类型转换成高级数据类型（表 2-1-6 中 double 比 float 高级，float 比 long 高级，long 比 int 高级，……）。

表 2-1-6 以 x*y 操作为例来说明 Java 中类型的自动转换。

表2-1-6 类型的自动转换

操作数x 的类型	操作数y 的类型	转换后类型
byte 或 short	int	int
byte 或 short 或 int	long	long
byte 或 short 或 int 或 long	float	float
byte 或 short 或 int 或 long 或 float	double	double
char	int	int

高级数据类型不能自动向低级数据类型转换，若需要进行这种转换，必须使用操作符“( )”来操作，才能把高级数据类型强制转换为低级数据类型。例如：

```
float x=3.1415f;
int y=(int)x;          // 经过转换，y 值为 3
int n=(int)(x*2);      // 对表达式进行强制转换时，要将表达式 x*2 用括号括起来
```

小资料 java的关键字

abstract	else	interface	switch
boolean	extends	long	synchronized
break	false	native	this
byte	final	new	throw
case	finally	null	throws
catch	float	package	transient
char	for	private	true
class	goto	protected	try
const	if	public	void
continue	implements	return	volatile
default	import	short	while
do	instanceof	static	
double	int	super	

图灵与“图灵奖”

图灵(Alan Turing,1912 ~ 1954),英国人,被认为是20世纪最著名的数学家和计算机理论家。1931年,图灵考入剑桥大学。1936年,24岁的图灵在伦敦权威的数学杂志上发表了《论数字计算在决断难题中的应用》一文,提出了著名的“图灵机”(Turing Machine)设想。后在其著名的论文《论可计算数在判定问题中的应用》中,论述了以布尔代数为基础,将逻辑中的任意命题用一种通用的机器来表示和完成,并能按照一定的规则推导出结论。图灵的这篇论文被誉为现代计算机原理的开山之作,它描述了一种假想的可实现通用计算的机器,这种机器被后人称为“图灵机”。图灵机理论不仅解决了纯数学基础理论问题,而且在理论上证明了研制通用数字计算机的可行性。1950年,他又发表了题为《机器能思考吗?》的论文,第一次提出“机器思维”的概念。后来人们称图灵为“人工智能之父”。

为了纪念图灵的贡献,美国计算机协会(Association for Computing Machinery,ACM)在1966年设立“图灵”奖,专门奖励那些对计算机科学研究与推动计算机技术发展有卓越贡献的杰出科学家。“图灵奖”被公认为计算机界的“诺贝尔奖”。

1. 下列可以作为合法变量名的是_____。

   A. a7　　B. 7a　　C. a-3　　D. public

2. 下面不合法的运算符是_____。

   A. +　　B. *　　C. ÷　　D. %

3. 在语句 int a[ ] 中,a 表示_____。

   A. 类名　　B. 数组名　　C. 数组元素　　D. 方法名

4. 表达式 (float)(2+3)/7 运算结果的数据类型是_____。

5. 下列程序段的执行结果是_____。

```
public class Array_1
{
    public static void main(String args[ ])
    {
        int a[ ] = {1, 2, 3}, b[ ];
        int c = 5, d;
        b = a;
        d = c;
        System.out.println(a[0]+ " " + a[1]+ " "+ a[2]+ " ");
        System.out.println(b[0]+ " " + b[1]+ " "+ b[2]+ " ");
        System.out.println("c=" + c + "    d=" + d);
    }
}
```

第二节　顺序结构

本节我们通过实例来学习顺序结构的程序设计方法，以及在程序设计中如何进行数据的输入与输出设计。

程序有三种基本控制结构，即顺序结构、分支结构和循环结构。顺序结构是三种结构中最基本的程序控制结构。

一　程序执行顺序

在顺序结构的程序中，算法的各个步骤是按语句的先后顺序执行的。

例如：在程序设计中经常需要交换两个变量的值，那么怎样实现呢？

变量的值是存放在内存单元中的。根据变量存储数据的这一特点，变量初始化后，若给变量多次赋值，那么该变量只能保留最后的值。所以，要交换两个变量中的数据，就要借助第三个变量（也称中间变量）来实现。

我们用 a，b 分别表示已经存放数据的变量，设 a，b 分别存放数据 3 和 5。用 t 表示中间变量，作为暂存数据用。交换 a，b 两个变量值的操作步骤如下：

第一步：把 a 的值 3 赋给 t，则 a，t 的值皆为 3；

第二步：把 b 的值 5 赋给 a，则 b，a 的值皆为 5；

第三步：把 t 的值 3 赋给 b，则 t，b 的值皆为 3。

此时 a，b 的值分别为 5 和 3。

上述过程如图 2-2-1 所示。

其算法可用图 2-2-2 所示的流程图表示。

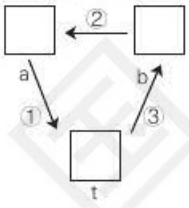


图2-2-1　交换a，b两个变量值的示意图

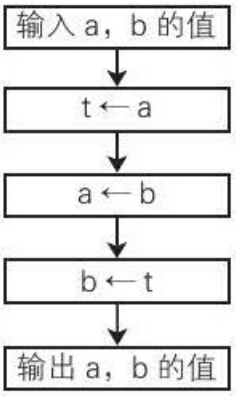


图2-2-2　交换a，b两个变量值的流程图

这是一种典型的顺序结构算法，实现上述算法的程序如下：

程序：ab.java

功能：交换两个变量 a，b 的值

```
public class ab
{
   public static void main(String args[ ])
   {
      int a = 3;                          // 声明并初始化变量 a
      int b = 5;                          // 声明并初始化变量 b
      int t;                              // 声明变量 t
```

```
        System.out.println( "a= "+a+" b="+ b); // 输出变量 a，b 的初值
        // 交换变量 a，b 的值
        t=a;                                    // 将 a 的值赋给 t
        a=b;                                    // 将 b 的值赋给 a
        b=t;                                    // 将 t 的值赋给 b
        System.out.println( "a= "+a+" b="+ b); // 输出 a，b 的等值
    }
}
```

我们进一步分析程序中的语句：

```
public static void main(String args[ ])
```

修饰符 public 声明方法 main( ) 是公共的，可被任何程序访问，static 声明方法 main( ) 是静态的，void 声明 main( ) 不返回任何值；参数 String args[ ] 声明了一个 String 型数组 args。

现在在“命令提示符”窗口中编译并运行程序。

首先，利用 javac 命令编译程序，输入：

```
javac ab.java
```

然后，利用 java 命令运行程序，输入：

```
java ab
```

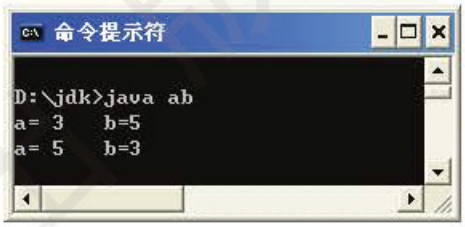


图2-2-3　程序ab的运行结果

运行结果如图 2-2-3 所示，可以看出变量 a，b 的值已经交换。

注：①编写 Java 程序时，特别要注意英文字母的大小写，否则将引起错误。

②中文标点符号只能出现在字符串和注释中。

③不要混淆小写英文字母 l 和数字 1 及字母 o 和数字 0。

二　输入与输出

设计程序与设计算法一样，一般分为三个步骤，如图 2-2-4 所示。

程序首先需要解决输入数据和输出结果的问题。在 Java 中，可以通过不同方式输入数据和输出结果。

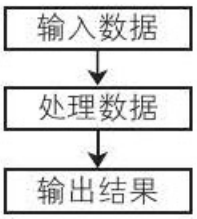


图2-2-4　设计程序的流程图

在上述程序中，直接对变量 a，b 赋值，这种方式比较简单。但是，如果事先不知道变量 a，b 的值，就无法直接提供数据。例如，计算任意两个正整数的和，首先就需要解决这两个正整数的输入问题。

1. 命令行方式输入数据

运行程序时，可以在命令行方式中给出参数，参数之间用空格分隔，这些参数将传递给 main( ) 方法的 args 数组。

例如：java Sum 3 5

参数 3 和 5 被传递给 main( ) 方法中的 args 数组。第一个参数传递给数组元素 args[0]，第二个参数传递给数组元素 args[1]。

在 Java 中，利用命令行方式输入的数据都是字符串型的，需要转换为数值型才能参与

算术运算。可以利用 Integer.parseInt( ) 方法将字符串转换为整数，利用 Double.parseDouble( ) 方法将字符串转换为浮点数。Java 提供的将字符串型数据转换为数值型数据的常用方法见表 2-2-1。

表2-2-1 字符串型数据转换为数值型数据的常用方法

方 法	功 能	实 例
Integer.parseInt(s)	把数字组成的字符串 s 转换为 int 型	Integer.parseInt("123") 的结果为 123
Long.parseLong(s)	把数字组成的字符串 s 转换为 long 型	Long.parseLong("999") 的结果为 999
Float.parseFloat(s)	把数字组成的字符串 s 转换为 float 型	Float.parseFloat("1.23") 的结果为 1.23
Double.parseDouble(s)	把数字组成的字符串 s 转换为 double 型	Double.parseDouble("3.1415") 的结果为 3.1415

例如两个变量互换程序 ab.java 中的语句：

```
int a=3;
int b=5;
```

可以修改为：

```
int a = Integer.parseInt(args[0]);
int b = Integer.parseInt(args[1]);
```

计算两个正整数和的程序代码如下：

程序：Sum2.java

功能：计算两个整数的和

```
public class Sum2                                    //Sum2 为程序文件名
{
   public static void main(String args[ ])          // 程序执行的起点
      {
         // 把从命令行接受的第一个参数赋值给整型变量 a
         int a = Integer.parseInt(args[0]);
         // 把从命令行接受的第二个参数赋值给整型变量 b
         int b = Integer.parseInt(args[1]);
         int c;          // 声明 c 为整型变量
         c=a+b;          // 计算两个数的和
         System.out.println(" 计算两个数的和 :");   // 输出字符串
         System.out.println(a+" + "+b+" = "+c);     // 输出运算结果
      }
}
```

程序经编译后，在命令行方式下输入：

```
java Sum2  3  5
```

其中 3 和 5 为输入的参数。运行结果如图 2-2-5 所示。

图2-2-5 程序Sum2 的运行结果

2. 从文本数据流读取数据

大多数情况下，特别是在不知道输入几个数据时，利用命令行方式输入数据就很不方便。不过，Java 还

提供了其他输入方式。例如：使用缓冲区从文本数据流中读取数据。

程序：Sum3.java

功能：计算两个浮点数的和

```
import java.io.*;
public class Sum3                                    // Sum3 为程序文件名
{
   public static void main(String args[ ])  throws  IOException
   {
      // 使用缓冲区 (BufferedReader) 从文本数据流读取数据
      InputStreamReader  reader=new  InputStreamReader(System.in);
      BufferedReader  input=new  BufferedReader(reader);
      System.out.print("Enter  the  a:");          // 提示输入 a 的值
      String  s1=input.readLine( );                // 从键盘输入第一个数字（字符串）
      double  a=Double.parseDouble(s1);            // 将数字 s1 转换为 double 型数 a
      System.out.print("Enter the b:");            // 提示输入 b 的值
      String  s2=input.readLine( );                // 从键盘输入第二个数字 ( 字符串 )
       double  b=Double.parseDouble(s2);           // 将数字 s2 转换为 double 型数 b
       double  c=a+b;                              // 计算两个数的和
       System.out.println(a+" + "+b+" = "+c);      // 输出运算结果
   }
}
```

代码中“import java.io.*;”用来导入 Java 处理数据流所需要的类“java.io”，其中“.*”表示可以使用 java.io 类中的所有类。

使用文本数据流输入数据时，可能会产生错误。Java 把运行程序时产生的错误称为异常（Exception）。为了避免因发生异常而导致失败，需要在 main( ) 的后面加上“throws IOException”，表示它将不处理异常（也称抛出 IO 异常）。程序编译后，运行结果如图 2-2-6 所示。

图2-2-6　程序Sum3的运行结果

3. 输出信息

在 Java 中常使用 System.out.print( ) 方法来输出各种信息。这里，System 是 Java 的类。常用的输出信息的方法如下：

```
System.out.print(x);                  // 输出变量 x 的值，不换行
System.out.print("abc");              // 输出字符串 abc（双引号中的内容），但不换行
System.out.println("abc");            // 输出字符串 abc（双引号中的内容），并换行
System.out.println( );                // 输出空行（相当于换行）
System.out.println("123"+"456");      // 输出字符串 123456，"+" 号表示接连两个字符串，并换行
```

三 应用实践

任务1 计算圆的面积。

①问题分析

用 r 表示圆的半径，area 表示圆的面积，由数学公式可知 area=πr^2。

由于问题中没有给出圆的半径的值，所以程序运行后需要通过键盘输入半径。

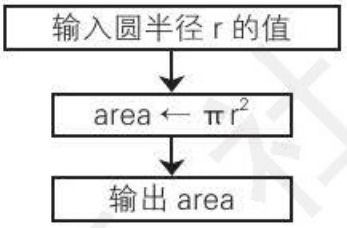


图2-2-7 计算圆面积的流程图

②算法流程

计算圆的面积的算法流程图如图 2-2-7 所示。

③编程实现

程序：AreaOfCircle.java

功能：计算圆的面积

```
import java.io.*;
public class AreaOfCircle                          // AreaOfCircle 为程序文件名
  {
    public static void main(String args[ ]) throws  IOException
      {  // 使用缓冲区 (BufferedReader) 从文本数据流读取文本数据
         InputStreamReader  reader=new  InputStreamReader(System.in);
         BufferedReader input=new  BufferedReader(reader);
         System.out.print(" 请输入半径 :");
         String  s=input.readLine( );                  // 从键盘输入一个字符串
         double  r=Double.parseDouble(s);              // 将字符串 s 转换为 double 型
         double  area=Math.PI*r*r;                     // 计算圆的面积
         System.out.println(" 圆的面积 : "+area);
      }
  }
```

程序中 Math.PI 是系统定义的一个常数，表示圆周率，它和其他一些常用数学函数都封装在 Math 类中。Math 类中常用的方法如表 2-2-2 所示。

表2-2-2 Math类常用方法

方法名	功能	举例
Math.abs(x)	返回 x 的绝对值	Math.abs(-8.5) 的值为 8.5
Math.ceil(x)	返回不小于 x 的最小整数的 doubl 型	Math.ceil(-8.3) 的值为 -8.0
Math.floor(x)	返回不大于 x 的最大整数的 doubl 型	Math.floor(-8.3) 的值为 -9.0
Math.round(x)	返回 x 四舍五入后的整数	Math.round(8.5) 的值为 9
Math.max(x,y)	返回 x,y 中的较大数	Math.max(2,3) 的值为 3
Math.min(x,y)	返回 x,y 中的较小数	Math.mix(2,3) 的值为 2
Math.random( )	返回 [0,1.0] 之间的随机数	0 ≤ Math.random( )<1

续表2-2-2

方法名	功能	举例
Math.pow(x,y)	幂运算，求 x^y	Math.pow(2,3) 的值为 8
Math.sqrt(x)	返回 x 的算术平方根	Math.sqrt(2) 的值为 1.4142135623730951
Math.exp(x)	返回指数函数 e^x 运算	Math.exp(1) 的值为 2.718281828459045
Math.log(x)	返回对数函数 ln(x) 的值	Math.log(1) 的值为 0
Math.sin(x)	返回正弦函数 sin(x) 的值	Math.sin(Math.PI/6) 的值为 0.49999999999999994
Math.cos(x)	返回余弦函数 cos(x) 的值	Math.cos(Math.PI/3) 的值为 0.50000000000000001
Math.tan(x)	返回正切函数 tg(x) 的值	Math.tan(Math.PI/4) 的值为 0.99999999999999999
Math.PI	常数 π 的值	3.141592653589793
Math.E	常数 e 的值	2.718281828459045

Pascal语言

熟悉软件编程的人大概都知道“算法＋数据结构＝程序”这一著名公式，它是瑞士计算机科学家尼克劳斯·威茨（Niklaus Wirth）提出的，威茨因此在1984年获得图灵奖。

威茨1934年2月15日出生于瑞士北部离苏黎世不远的温特图尔（Winterthur）。威茨小时候就喜欢动手动脑，他的最大爱好就是组装飞机模型。中学毕业以后，威茨进入著名的瑞士苏黎世工学院（ETH），1958年取得学士学位。1963年获得美国加州大学伯克利分校博士学位。

1968年威茨回到母校苏黎世工学院。在这里，他成功设计出了Pascal语言。Pascal的许多特点后来被C，Ada等语言加以继承并发展。因此，Pascal在程序设计语言的发展史上具有承上启下的重要里程碑意义。

1971年，威茨基于开发程序设计语言和编程的实践经验，首次提出了“结构化程序设计”（Structured Programming）的概念。威茨提出的这种结构化程序设计方法又称为“自顶向下”或“逐步求精”法，成为程序开发者共同遵守的一个标准方法，在后来的软件工程中也得到广泛应用。

20世纪70年代中期，为适应并发程序设计的需要，威茨又成功开发了Modula语言并得到广泛应用。Modula除了提供并发程序设计功能之外，另一个重要特征就是引进了模块概念（这也是这个语言命名为Modula的原因）以及“进程”（Process）这一和并发程序相联系而产生的重要概念。

1．已知自由落体的位移公式为：$s=\frac{1}{2}gt^2+v_0t$，其中 v_0 为初始速度，g 为重力加速度，t 为经历的时间。编程求位移。

2．输入圆柱体的半径和高，计算并输出圆柱体的体积和表面积。

3．查找有关地球的数据资料，计算地球的体积和表面积，以及大气层的容积等。

第三节 分支结构

电子计算机与计算器的区别之一就在于计算机具有逻辑判断能力，它通过条件判断来决定程序执行的分支结构的路径。本节我们将学习简单分支结构和多分支结构的程序设计方法。

程序在执行到某一个步骤时，可能需要对当时的处理结果进行判断，然后根据判断结果再去执行不同的程序块。在程序设计中，可以使用分支结构来实现判断。分支结构分为简单分支结构和多分支结构，Java 语言分别用 if 语句和 switch 语句来实现。

一 简单分支结构

判断一个正整数奇偶性的方法通常是：如果一个正整数能被 2 整除，那么这个数是偶数，否则是奇数。

上述问题的算法用自然语言描述如下：

输入：一个正整数 x
输出：x 是奇数或 x 是偶数
指令：
1. 输入一个正整数 x；
2. 如果 x 能被 2 整除，那么 x 为偶数；
否则 x 为奇数。

该算法还可以用框图表示，如图 2-3-1 所示。其中，T 表示条件成立（true），F 表示条件不成立（false）。

在 Java 语言中，简单分支结构用 if 语句来实现，格式如下：

```
if( 条件表达式 )
  {
     语句或语句块；
  }
else
  {
     语句或语句块；
  }
```

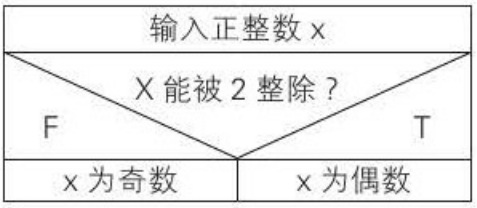


图2-3-1 判断正整数奇偶性的框图

条件表达式的值为逻辑型(true 或 false),当条件表达式的值为 false 时,不需要做任何事,这时 else 部分可被省略。

为便于阅读，编写程序时通常采用缩进的形式书写语句或语句块。if 语句中，语句块哪怕只有一行语句，一般也要用花括号括起来，这样可增强程序的可读性。

判断正整数奇偶性问题，用 if 语句编写的程序代码如下：

程序：if1.java

功能：判别一个正整数的奇偶性

```
public class if1
{
    public static void main(String args[ ])
    {
      // 把从命令行方式接受的第一个参数赋给整型变量 x
      int x = Integer.parseInt(args[0]);
      if (x % 2 ==0 )                                    // 如果 x 能被 2 整除
      {
            System.out.println(x+" 是偶数。");   // 那么输出 x 是偶数
      }
      else
      {
            System.out.println(x+" 是奇数。");   // 否则输出 x 是奇数
      }
    }
}
```

其中，“x%2==0”叫做条件表达式。

分支结构还可以使用条件运算符“?:”来实现，条件运算符“?:”是个三目运算符，可以实现简单的判断功能。

格式：表达式 1? 表达式 2: 表达式 3;

如果表达式 1 的值为 true，则取表达式 2 的值，否则取表达式 3 的值。

例如：

```
max=(a>b)? a:b;
```

表示“如果 a>b，那么把 a 的值赋给 max，否则把 b 的值赋给 max”。

二　条件语句的嵌套

if-else语句可以嵌套。即在一个if结构中，可以内嵌一个完整的if结构。

例如：判断一个数是正数、零还是负数。其算法可以用框图描述，如图 2-3-2 所示。

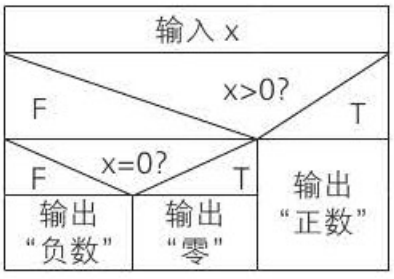


图2-3-2　判断正负数的框图

根据框图很容易写出 Java 程序。不过，如果你是通过命令行输入参数，那么在为程序输入数据时，很可能不知道应该输入几个参数。我们可以在程序中增加判断输入参数个数的代码段。

```
if (args.length ! = 1)
// 如果输入的参数个数不是 1，提示输入一个数
```

```
{
    System.out.println(" 请输入一个数！ ");    // 提示输入一个参数
    return;                                    // 使用语句 return 退出 main( ) 方法
}
```

完整的程序如下：

程序：if2.java

功能：判断一个数是正数、零还是负数

```
public class if2
{
  public static void main(String args[ ])
  {
    if(args.length ! = 1)      // 如果输入的参数个数不是 1，提示输入一个数
     {
       System.out.println(" 请输入一个数！ ");
       return;                 // return 表示退出 main( ) 方法
     }
    // 把从命令行接受的第一个参数赋值给整型变量 x
    int x = Integer.parseInt(args[0]);
    if (x >0 ) System.out.println(x+" 是正数。");
    else
     {
       if (x==0) System.out.println(x+" 是零。");
       else System.out.println(x+" 是负数。");
     }
  }
}
```

三 多分支结构

if 语句只能实现简单分支结构。Java 提供了实现多分支结构的 switch 语句。它的格式如下：

```
switch ( 表达式 )
{
  case 表达式 1:
       语句块 1;
       break;
  case 表达式 2:
       语句块 2;
    ⋮
```

```
            break;
        default:
            语句块 3;
    break;
}
```

它的执行过程是：在进入 switch 结构之前，程序先计算出表达式的值。进入 switch 之后，用该值依次与 case 中每一个表达式的值进行比较，如果匹配，就执行 case 后的语句块，经 break 语句结束 switch 结构；如果都不匹配，就执行 switch 结构中 default 后的语句块，如果没有 default 语句，就去执行 switch 结构之后的第一个可执行语句。

在 case 语句块中，一般都设置一个 break 语句作为 switch 结构的出口。当不设置 break 语句时，就继续执行下一个 case 中的第一个可执行语句，以此来满足某种特殊需要。

在“switch(表达式)”语句中，“表达式”的值必须是 int 类型或可以转换成 int 类型的原始数据类型，不允许使用 float，double，long 或 String 等数据类型。

下面的程序是用 switch 语句编写的，当输入任意一个月份时，输出该月份属第几季度。

程序：jidu.java

功能：当输入任意一个月份时，输出该月份属第几季度

```
// 当输入任意一个代表月份的数字时，输出该月份属第几季度
import java.io.*;
public class jidu
{
    public static void main(String args[ ]) throws IOException
    {
        // 使用缓冲区 (BufferedReader) 从文本数据流读取文本数据
        InputStreamReader reader=new InputStreamReader(System.in);
        BufferedReader input=new BufferedReader(reader);
        System.out.print(" 请输入 1~12 之间的一个整数 :");
        String s=input.readLine( );     // 从键盘输入一个字符串
        int m=Integer.parseInt(s);      // 将字符串 s 转换为整数
        int n=(m-1) / 3;                // 将 m 的值转换为季度数 n
        switch (n)
        {
          case 0:    // 当 n=0 时，则输出“第一季度”
                System.out.println(m+" 月属第一季度 ");
                break;
          case 1:    // 当 n=1 时，则输出“第二季度”
                System.out.println(m+" 月属第二季度 ");
                break;
          case 2:    // 当 n=2 时，则输出“第三季度”
                System.out.println(m+" 月属第三季度 ");
                break;
```

```
            case 3:    // 当 n=3 时，则输出“第四季度”
                System.out.println(m+" 月属第四季度 ");
                break;
        }
    }
}
```

四 应用实践

任务 1

计算汇款资费。

①问题分析

邮局计算汇款资费的方法如下：汇款金额小于 100 元，汇费为 1 元；汇款金额在 100 元到 5 000 元之间，汇费按 1% 收取；汇款金额大于 5 000 元，汇费为 50 元。试编程计算汇款 x 元所需的汇费。

②算法描述

计算汇款资费问题的框图如图 2-3-3 所示。

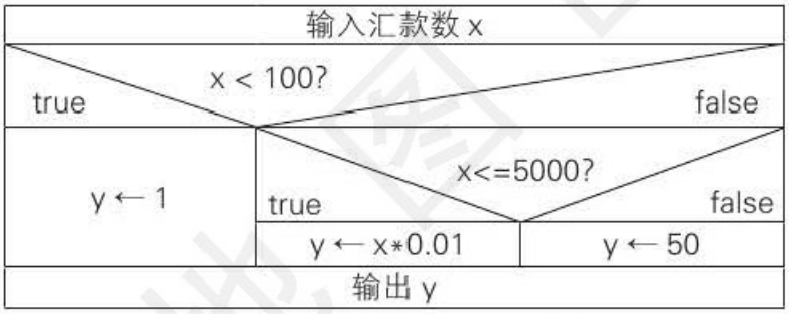


图2-3-3 计算汇款资费问题的框图

③程序实现

程序：Huikuan.java

功能：计算汇款资费问题

```
//计算汇款资费问题
import java.io.*;                          // 导入所需要的公共类
public class Huikuan
{
   public static void main(String args[ ]) throws IOException
   {
      // 使用缓冲区 (BufferedReader) 从文本数据流读取文本数据
      InputStreamReader reader=new InputStreamReader(System.in);
      BufferedReader input=new BufferedReader(reader);
      System.out.print(" 请输入汇款金额：");
      String s1=input.readLine( );                 // 从键盘输入一个字符串
      double x=Double.parseDouble(s1);             // 将字符串 s1 转换为浮点数
```

```
        double y;                          // 声明变量 y, 保存汇费
        if (x < 100)                       // 如果汇款金额 x<100，则汇费为 1 元
        {
           y = 1.0;
        }
        else if (x<=5000)
        {
           y = x*0.01;      // 如果汇款金额 100 ≤ x ≤ 5000，则汇费为 x*0.01 元
        }
        else
        {
           y = 50.0;        // 如果汇款金额 x>5000，则汇费为 50 元
        }
        System.out.println(" 汇款金额 : " + x );
        System.out.println(" 汇费 :      " + y );
    }
}
```

2　为“我最喜爱的体育项目”投票。

①问题的提出

我们拟设计一个“我最喜爱的体育项目”的投票活动。这里只设计投票程序的输入子模块，随着学习的深入，我们将不断完善投票程序。

约定各体育项目的代码：足球为 1，篮球为 2，排球为 3，乒乓球为 4，羽毛球为 5，田径为 6，游泳为 7，跳水为 8，体操为 9，其他为 0。

请用分支结构语句设计这个投票程序，当分别输入 1 ~ 9 之间的数字时，输出你最喜爱的体育项目。

②算法流程

投票程序的框图如图 2-3-4 所示。

<table>
<tr><td colspan="10">输入 n</td></tr>
<tr><td colspan="10">switch (n)</td></tr>
<tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td></tr>
<tr><td>输出
足球</td><td>输出
篮球</td><td>输出
排球</td><td>输出
乒乓球</td><td>输出
羽毛球</td><td>输出
田径</td><td>输出
游泳</td><td>输出
跳水</td><td>输出
体操</td><td>输出
其他</td></tr>
</table>

图2-3-4　投票程序的框图

③编程实现

程序：vote.java

功能：当分别输入 1 ~ 9 时，输出你最喜爱的体育项目

```
// 当分别输入 1~9 时，输出你最喜爱的体育项目
```

```
import java.io.*;                    // 导入所需要的公共类
public class vote
{
  public static void main(String args[ ]) throws IOException
  {
 System.out.println(" 足球为 1，篮球为 2，排球为 3，乒乓球为 4，羽毛球为 5");
 System.out.println(" 田径为 6，游泳为 7，跳水为 8，体操为 9，其他为 0");
 // 使用缓冲区 (BufferedReader) 从文本数据流读取文本数据
 InputStreamReader reader=new InputStreamReader(System.in);
 BufferedReader input=new BufferedReader(reader);
 System.out.print(" 请输入你喜爱的体育项目编号（0~9）:");
 String s=input.readLine( );     // 从键盘输入数字 0~9
 int n=Integer.parseInt(s);      // 将字符串 s 转换为整数 n
 switch (n)
 {
    case 1:
          System.out.println(" 你喜爱的体育项目是足球！ ");
          break;
    case 2:
          System.out.println(" 你喜爱的体育项目是篮球！ ");
          break;
    case 3:
          System.out.println(" 你喜爱的体育项目是排球！ ");
          break;
    case 4:
          System.out.println(" 你喜爱的体育项目是乒乓球！ ");
          break;
    case 5:
          System.out.println(" 你喜爱的体育项目是羽毛球！ ");
          break;
    case 6:
          System.out.println(" 你喜爱的体育项目是田径！ ");
          break;
    case 7:
          System.out.println(" 你喜爱的体育项目是游泳！ ");
          break;
    case 8:
```

```
            System.out.println(" 你喜爱的体育项目是跳水！ ");
            break;
        case 9:
            System.out.println(" 你喜爱的体育项目是体操！ ");
            break;
        default:            // 否则输出其他
            System.out.println(" 这里没有你喜爱的体育项目！ ");
            break;
        }
    }
}
```

算法与数字生活

算法与数字生活息息相关，如使用机器人清理房间、通过智能运动手环进行科学健身、输入用户口令安全登录账户等，算法都在其中起着重要的作用。借助高效的算法和优化的程序，我们可以更加快捷方便地解决越来越多的问题，提高生活、工作和学习效率。

用数字化工具解决问题离不开程序的控制，而程序的核心是算法设计。算法描述了解决问题的步骤，程序则是用某种程序设计语言将设计好的算法进行具体实现，从而实现自动化的问题求解。

例如，空调具有的定时关机功能，就可以通过倒计时算法来实现。在设定关机时间后，系统将进行倒数计数，当计数结束时就会发出关机指令，关闭空调。

1．当给出 x 的值时，求下列函数的值。

$$y=\begin{cases}0 & (X<0)\\ \sqrt{X} & (X>=0)\end{cases}$$

2．编写程序，输入年份，判断该年是否闰年。

3．某商场为了促销苹果，规定购买 2 千克以上可以在原价每千克 1.5 元的基础上打 8 折。请设计一个程序计算购买 x 千克苹果的应付款。

第四节 循环结构

循环是程序设计的基本结构之一。本节通过学习 while 语句、do...while 语句以及 for 语句，掌握计数循环和条件循环的不同实现方法，并练习使用循环语句设计解决实际问题的程序。

程序中经常需要重复执行某些程序段，这可以使用循环语句来实现。被重复执行的程序段称为循环体。循环一般是有条件的，即在满足一定条件下，才能执行循环体，或在满足一定条件下，不再执行循环体。

一 循环语句的结构

一个循环语句一般应包括以下内容：

◆初始化部分。用来设置循环的一些初始条件。

◆循环体部分。这是重复循环的一段代码，可以是单条语句，也可以是多条语句（语句块）。

◆终止部分。通常是一个逻辑表达式，进入一次循环前要对该表达式求值。如果该值为真，就进入此次循环，否则将终止整个循环。

在日常生活中，我们经常会遇到类似“求任意若干个数（例如 1，3，5，6，9，五个数）的和”的问题。解决此类问题的一般步骤如下：

操作步骤	操作方式	操作结果显示
操作前	接通计算器电源	0
第 1 步	输入数 1 输入运算符 +	1 1
第 2 步	输入数 3 输入运算符 +	3 4
第 3 步	输入数 5 输入运算符 +	5 9
第 4 步	输入数 6 输入运算符 +	6 15
第 5 步	输入数 9 输入运算符 = 或 +	9 24

从以上步骤可以看出，每次加法运算都是在求出和的基础上进行的。每一步操作方式都是相同的，或者说是重复相同的操作，因此可以用循环语句来实现。

这种循环过程，可以用下面两个不同的流程图表示，如图 2-4-1。

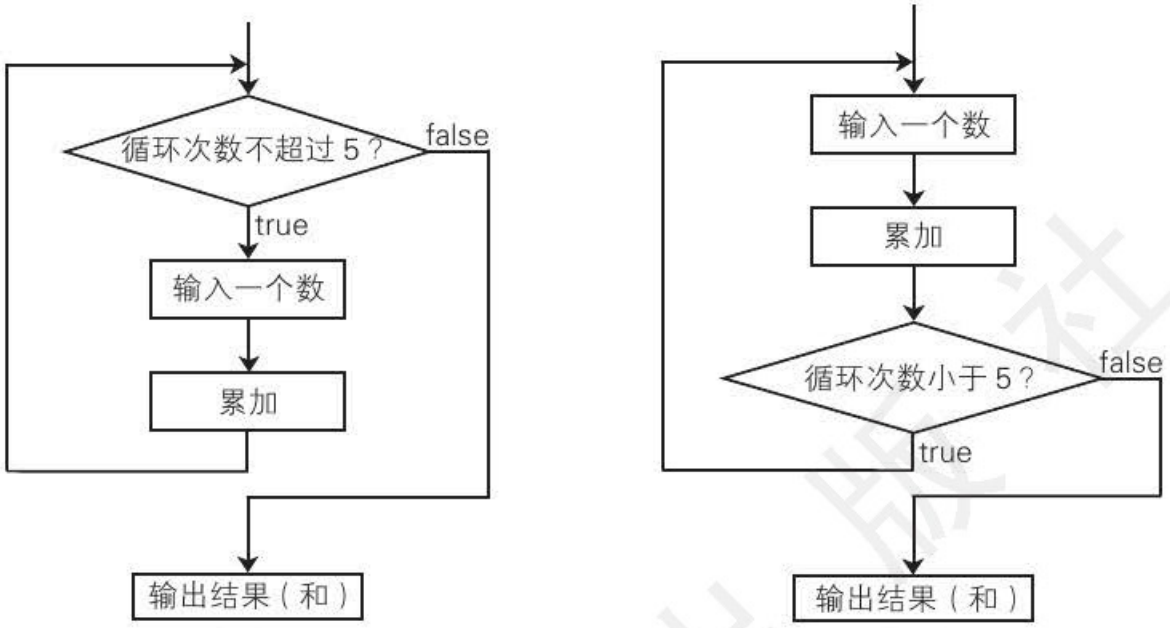


图2-4-1　循环结构流程图

这两个流程图的区别仅在于判断循环条件的位置不同，图 2-4-1（a）表示当条件表达式为真时，执行循环，所以称为当型循环；图 2-4-1（b）表示先执行一段语句，然后判断条件表达式的值，如果值为真，则继续循环，直到条件表达式为假，所以称为直到型循环。

上述过程用框图表示如图 2-4-2，图 2-4-2（a）为当型循环，图 2-4-2（b）为直到型循环。

图2-4-2　循环结构的框图

Java 有三种语句可以实现循环结构的设计，分别是 while 循环、do……while 循环和 for 循环。

1. while循环

while 循环的格式如下：

```
while( 条件表达式 )
  {
     循环体；
  }
```

while 循环是一种条件循环。while 语句首先计算条件表达式的值，当值为 true 时，执行循环体，否则就跳出循环语句，这是当型循环的特点。在循环体内，一定要有改变条件表达式值的语句，以防止死循环（即循环无法终止）。

求任意若干个数的和也可以使用 while 语句来实现。其中循环部分代码如下：

```
int i=0 ;                          // 声明并初始化循环变量 i
while (i<5)                        // 设置循环条件
  {
     System.out.print(" 请输入一个数 :");
     c=input.readLine( );          // 从键盘输入一个数
     x=Double.parseDouble(c);      // 将字符串 s 转换为数值 ;
     s=s+x;                        // 累加求和
     i++;                          // 计数，改变循环条件
  }
```

2. do…while循环

do…while 循环的格式如下：

```
do
  {
     循环体 ;
  } while ( 条件表达式 ) ;
```

do…while 循环也是一种条件循环。do…while 语句首先执行循环体，然后计算条件表达式的值，如果值为 true，则继续执行循环体，直到值为 false，就跳出循环。与 while 语句不同的是，do…while 语句的循环体至少执行一次，这是直到型循环的特点。另外，在循环体内，也要有改变循环条件的语句。

求任意若干个数的和也可以使用 do…while 语句来实现。其中循环部分代码如下：

```
int i=0;
do
     {
        System.out.print(" 请输入一个数 :");
        c=input.readLine( );          // 从键盘输入一个字符串
        x=Double.parseDouble(c);      // 将字符串 c 转换为数值
        s=s+x;                        // 累加求和
        i++;                          // 计数，改变循环条件
     } while (i<5);                   // 设置循环条件
```

3. for循环

for 循环的格式如下：

```
for ( 循环变量初始化 ; 终止条件表达式 ; 循环变量增量 )
  {
     循环体 ;
  }
```

for 语句执行时，首先初始化循环变量，然后判断终止条件表达式的值，如果值为 true，则执行循环体，然后按照增量更新循环变量的值。完成一次循环后，重新判断终止条件表达式的值，如果值为 false，则退出循环语句。for 语句是一种计数循环。

例如，“for(i=0; i<10; i++)”语句，它将 i 值初始化为零，并在每执行完一次循环体后，按增量 1 增加 i 的值，当 i=10 时退出循环。

利用 for 语句实现求 5 个数之和的 Java 程序如下：

程序：Sum4.java

功能：用 for 语句实现求 5 个数的和

```
import java.io.*;
public class Sum4
{
    public static void main(String args[ ]) throws IOException
    {
        double x,s=0;
        String c;
        // 使用缓冲区 (BufferedReader) 从文本数据流读取文本数据
        InputStreamReader reader=new InputStreamReader(System.in);
        BufferedReader input=new BufferedReader(reader);
        for (int i=0;i<5;i++)                    // 循环 5 次
        {
           System.out.print(" 请输入一个数 :");
           c=input.readLine( );                  // 从键盘输入一个字符串
           x=Double.parseDouble(c);              // 将字符串 c 转换为数值
           s=s+x;                                // 累加求和
        }
        System.out.println(" 所求和 s="+s);
    }
}
```

我们在程序设计中应该根据不同的需要选择不同的循环语句。for 循环是一种计数循环，它适用于已知循环次数的循环结构；while 循环与 do…while 循环适用于循环次数事先不可确定的情况，或者用于处理各种条件控制的循环。

创建一个包含n个任意两位正整数的数组，输出该数组，并求这n个两位正整数的平均值。

①算法分析

第一步：输入正整数的个数 n。

第二步：初始化数组。

在使用一个数组之前，必须先对此数组进行声明：

```
类型  数组名[];  或 类型[] 数组名;
```

现在使用数组 a 来存放 n 个正整数，因此声明如下：

```
int a[];  // 声明一个整型一维数组
```

为了能使用数组，必须对数组初始化，在内存中留出相应的存储空间。数组初始化有两种方法：

一是直接初始化。直接初始化是在数组声明时进行的。

例如：

```
int b[]={1,3,5,7,9};  // 声明并初始化一个长度为 5 的整型数组
```

数组长度即数组元素的个数。

二是动态初始化。当事先不知道数组元素的值时，就需要使用动态初始化，即使用 new 操作符来分配内存空间。格式如下：

```
类型  数组名[];                    // 声明一个数组
数组名 = new 类型[数组长度]; // 初始化
```

或简写为：

```
类型  数组名[]= new 类型[数组长度]
```

现在对数组 a 进行动态初始化：

```
int a[]= new int[n];  // 声明并初始化数组 ,n 为数组 a 长度
```

第三步：为数组赋值。

声明和初始化一个数组以后，就可以通过数组名和下标来访问数组元素。数组的下标相当于一个指针，它指出了该元素在数组中的位置。一般使用循环的控制变量来访问数组的全部或部分元素。

在 Java 中，用来指示单个数组元素的最小下标是 0，使用时须保持在合法范围之内——大于等于 0 并小于数组长度，否则运行时会出现错误。

数组长度可以使用 String 类的 length( ) 方法获得，例如 a.length( ) 可以获得数组 a 的长度。

利用 Math.random( ) 方法生成两位随机正整数的表达式可表示为：

```
(int)(Math.random( )*90+10);
```

可以利用计数循环（for）来给数组元素 a[i] 赋值：

```
for (int i = 0; i < n; i++)              // 生成 n 个随机数
  {
      a[i]= (int) (Math.random( )*90+10);
  }
```

第四步：输出数组。

将数组中的数据输出，可以使用 System.out.print( ) 方法，也可以用计数循环（for）的方法。

```
for (int i = 0; i < n; i++)
   {
```

```
        System.out.print(a[i]+ " ");
    }
```

第五步：数组元素求和。

```
for (int i = 0; i < n; i++)
  {
        sum=sum+a[i];    // 累加
  }
```

第六步：求平均值，average=sum/n。

第七步：输出计算结果。

②程序实现

```
// 利用随机数生成一个包含 n 个两位正整数的数组，并求平均值
import java.io.*;                          // 导入所需要的公共类
public class Array1
{
    public static void main(String args[ ]) throws IOException
    {
        double sum=0;                      // 声明并初始化变量 sum，用来存放和
        // 输入正整数个数（n）
        // 使用缓冲区 (BufferedReader) 从文本数据流读取文本数据
        InputStreamReader reader=new InputStreamReader(System.in);
        BufferedReader input=new BufferedReader(reader);
        System.out.println(" 输入正整数个数 (n):" );   // 显示输入信息
        String s=input.readLine( );        // 从键盘输入一个字符串
        int n=Integer.parseInt(s);         // 将字符串 s 转换为整数
        int a[ ] = new int[n];             // 声明并初始化数组 a
        for (int i = 0; i < n; i++)        // 生成 n 个两位随机正整数
          {
            a[i]=(int)(Math.random( )*90+10);
          }
        for (int i = 0; i < n; i++)        // 输出数组 a 的各元素
          {
            System.out.print(a[i]+ "  ");
          }
        System.out.println( );
        for (int i = 0; i < n; i++)        // 计算数组 a 中 n 个元素的和
          {
            sum=sum+a[i];                  // 累加
          }
```

```
        double average=sum/n;                                    // 计算平均值
        System.out.println(n+" 个正整数的平均值为 : " + average);  // 输出结果
    }
}
```

二维数组

只有一个下标的数组叫做一维数组。例如：int a[ ] 表示 a 是一维数组。

数组的元素也可以是数组。如果一个数组的每一个元素都是一个一维数组，那么就构成了一个二维数组。二维数组的声明和初始化与一维数组相似。例如，int d[ ] [ ] 表示 d 是二维数组。

访问一维数组可以用单循环来实现，访问二维数组则要用二重循环来实现。

二 循环嵌套

一个循环结构内可以含有另一个循环，这样的结构称为循环嵌套，又称多重循环。常用的循环嵌套是二重循环，外层循环称为外循环，内层循环称为内循环。

下面的代码采用了循环嵌套结构，程序运行结果可以显示循环变量值的变化情况。

```
public class for2
{
    public static void main(String args[ ])
    {
        for(int i=1;i<=2;i++)          // 声明并初始化循环变量 i，设置循环终值 i<=2
        {                              外循环
            for(int j=1;j<=3;j++)      // 声明并初始化循环变量 j，设置循环终值 j<=3
            {                          内循环
                System.out.println(i+"  "+j);  // 输出 i，j 的值
            }
        }
    }
}
```

从这个二重循环结构上看，内循环（for j）是外循环的循环体。它的执行过程仍要遵循循环语句的执行原则。首先执行外层循环，即 i 由 1 至 2 执行 2 次。外层循环每执行一次，内层则执行一个完整的循环，即执行 3 次。循环体（即输出 i、j 的值）共执行 6 次。

for 语句的初始化部分可以声明变量。例如，语句 for（int i=1; i<=2; i++）声明并初始化了循环变量 i，i 的作用范围为整个 for 语句。

程序经编译后，运行结果如下：

```
1       1
1       2
1       3
2       1
2       2
2       3
```

运行结果已经反映出循环变量 i 与 j 变化的对应关系。

使用循环嵌套时，内外层循环的层次要分清，外循环一定要套住内循环，不能交叉（即循环语句不能交叉），并且内层和外层的循环语句不能声明相同的循环变量名。

三　循环控制语句

在各种循环中，还可以使用特殊流程控制语句：break 和 continue。

break 语句用来终止某个循环，使程序跳到循环体以外的第一个可执行语句。continue 语句只能出现在循环体中，作用是结束当前循环进入下一次循环。

break 和 continue 语句的使用可以参看下面的程序：BreakLoop.java。

```
// 使用 break 和 continue
public class BreakLoop
{
    public static void main(String args[ ])
    {
        int num1=0;       // 初始化计数器 num1，记录第 1 个循环的循环次数
        int num2=0;       // 初始化计数器 num2，记录第 2 个循环的循环次数
        for(int i=1;i<5;i++)          // 进入循环
        {
            if (i==3) break;          // 当 i=3 时，退出循环
            System.out.println(" 第一个循环语句的 i= "+i);
            num1=num1+1;              //num1 计数
        }
        System.out.println(" 第一个循环的次数："+num1);  // 输出循环次数
        for(int i=1;i<5;i++)
        {
            if (i==3) continue;       // 当 i=3 时，继续下一次循环
            System.out.println(" 第二个循环语句的 i= "+i);
            num2=num2+1;      //num2 计数
        }
        System.out.println(" 第二个循环的次数："+num2);  // 输出循环次数
    }
}
```

程序的运行结果如下：

```
第一个循环语句的 i=1
第一个循环语句的 i=2
第一个循环的次数：2
第二个循环语句的 i=1
第二个循环语句的 i=2
第二个循环语句的 i=4
第二个循环的次数：3
```

四 应用实践

任务1 计算棋盘上的麦粒。

①问题的提出

在印度有一个古老的传说。国王舍罕打算奖赏国际象棋的发明人——宰相西萨·班·达依尔，问宰相想要什么。宰相对国王说："陛下，请您在这张棋盘的第1个小格里，赏给我1粒麦子，第2个小格给2粒，第3个小格给4粒，每一小格都比前一小格加一倍，就这样摆满棋盘上所有64格，就把这些麦粒都赏给您的仆人吧！"国王觉得这要求太容易满足了，就命令给他这些麦粒。当人们把一袋一袋的麦子搬来开始计数时，国王才发现：就是把全印度甚至全世界的麦粒都拿来，也满足不了这位宰相的要求。那么，宰相要求得到的麦粒到底有多少呢？请你编写程序算一算共需要多少麦粒。

②算法分析

如表2-4-1所示，所需麦粒的总数为：

$s=2^0+2^1+2^2+2^3+\cdots+2^{63}$

即 $s_{i+1}=s_i+2^i$

上述规律可以使用循环语句实现，循环体中的重要语句为：

```
s=s+Math.pow(2,i)
```

i由0变化到63。其中，Math.pow(2,i)表示2^i。

表2-4-1 麦粒计算表

棋 盘	麦粒数	数学表达式
第1格	1	2^0
第2格	2	2^1
第3格	4	2^2
第4格	8	2^3
……	……	……
第64格	1.8481…E19	2^{63}

③编程实现

根据上述算法编写的程序如下：

```
//计算棋盘上的麦粒问题
public class Sum64
{
   public static void main(String args[ ])
      {
        double s=0;                    //声明并初始化变量s，存放麦粒数的累加和
```

```
        for (int i=0;i<64;i++)          // 循环 64 次
        {
        s=s+Math.pow(2,i);              // 累加
        }
        System.out.println("s="+s);  // 输出结果
      }
}
```

编译后，程序的运行结果为：

```
s=1.8446744073709552E19
```

可以看出，共需要 $1.8446744073709552\times10^{19}$ 粒麦子。大家可上网查一查，1 升麦子约合多少粒。那么，国王应该赏赐达依尔多少升麦子呢？全世界每年能生产多少麦子？要生产这么多的麦子需要多少年？

④算法优化。

$1+2^1+2^2+2^3+\cdots+2^{63}$ 的计算过程还可以表示为：

$$\begin{aligned} s &= 1+2^1+2^2+2^3+\cdots+2^{63} \\ &= 2^0+2^1+2^2+2^3+\cdots+2^{63} \\ &= 2^{63}+2^{62}+\cdots+2^3+2^2+2^1+2^0 \\ &= \underbrace{(\cdots((2+1)\times2+1)\times2+1)\times2+1}_{61\text{ 次}} \end{aligned}$$

上述规律可以表示为：$S_i=S_{i-1}\times2+1$，试根据这种算法重新编写程序。

2　处理字符串。

①问题的提出

将输入的字符串按逆序打印出来。

②算法分析

本题使用字符串类的方法来处理，大致需要以下三个步骤来完成。

第一步：输入字符串 s1。

例如，输入字符串 "Java" 给 s1。由于 Java 系统中的字符串实质是以数组形式存放的，所以 s1 也可以表示为 s1[0]='J'，s1[1]='a'，s1[2]='v'，s1[3]='a'。

第二步：声明一个空字符串 s2，使用逆序方式逐个将 s1 中的字符追加到空串 s2 中。

第三步：输出 s2，即将原输入的字符串按逆序打印出来。

③程序实现

```
import java.io.*;
public class string_4
{
    public static void main(String args[ ]) throws IOException
    {
```

```
        // 使用缓冲区 (BufferedReader) 从文本数据流读取文本数据
        InputStreamReader reader=new InputStreamReader(System.in);
        BufferedReader input=new BufferedReader(reader);
        System.out.println(" 输入一个字符串 :");    // 提示输入信息
        String s1=input.readLine( );                 // 从键盘输入一个字符串
        StringBuffer sa = new StringBuffer( );       // 创建 StringBuffer 的对象 sa
        String s2;
        // 逐个从输入的字符串 s1 中，逆序读入字符并追加到 sa 中
        int n= s1.length( );                         // 获得字符串 s1 的长度
        for (int i = (n – 1); i >= 0; i--)
        {    sa.append(s1.charAt(i));    // append( ) 方法功能是在 sa 后添加字符
        }
        s2 = sa.toString( );                         // 生成新的字符串 s2
        System.out.println(" 原字符串 : " + s1);
        System.out.println(" 逆字符串 : " + s2);
    }
}
```

程序编译后运行，结果如图 2-4-3 所示。

图2-4-3 字符串处理结果

构造字符串

在 Java 中，用两种方式构造字符串：

一种方式是直接将若干个字符用西文双引号括起来。例如：

```
String s="Java";
```

另一种方式是使用字符串类的方法。Java 为字符串操作提供了两个标准类，分别是 String 和 StringBuffer。

例如：String s; 声明 s 是 String 类，但并没有建立 String 的对象，只是建立一个 String 的引用 s，由于没有赋值，所以 s 的初始值为 null（空值）。需要利用 new 来建立 String 类的对象，例如：

```
String s=new String("ABC");
```

同样，可利用 new 来建立 StringBuffer 类的对象，例如：

```
StringBuffer s=new StringBuffer("ABC");
```

建立字符串对象之后，可以通过 String 类与 StringBuffer 类提供的方法对字符串进行操作。例如，可以使用 String 类的 length( ) 方法获取字符串长度，也还可以直接调用 String.ValueOf( ) 方法将数值类型转换为字符串：

```
float x=123.45f;
String s=String.ValueOf(x);
```

String 类和 StringBuffer 类的区别在于：String 类的字符串对象只能读取，而 StringBuffer 类的字符串对象除可以读取外，还可以改变其长度和内容。因此，如果要对一个空串追加字符，则这个空串必须声明为 StringBuffer 类。通常使用 StringBuffer.append( ) 方法向 StingBuffer 类的字符串尾端追加字符或字符串。

另外，可以通过调用 toString( ) 来显示一个对象的字符串。toString( ) 为 java.lang 包中 Object 类的 public 方法，常用的 String 类方法参见表 2-4-2。

表2-4-2 String类常用字符串的操作方法

方 法	功 能	举 例 String s="abcdefGH";
s.length( )	求字符串 s 中所含字符（包括空格）的个数（长度）	s.length( ) 的值等于 8
s.substring(b,e)	求出从第 b 个到第 e 个字符之间的字符组成的字符串 (b,e 的位置从 0 开始计算）	s.substring(2,3) 的值为 "c"
s.toUpperCase( )	将字母全部转换为大写	s.toUpperCase( ) 的值为 "ABCDEFGH"
s.toLowerCase( )	将字母全部转换为小写	s.toLowCase( ) 的值为 "abcdefgh"
s.concat(str)	连接字符串 s 与字符串对象 str	s.concat("Java") 的值为 "abcdefGHJava"
s.charAt(n)	求字符串 s 中第 n 个位置的字符	s.charAt(2) 的值为 c
s.indexOf(ch)	求字符串 s 中字符 ch 的第一个位置	s.indexOf('c') 的值为 2
s.indexOf(String s1)	求字符串 s 中子字符串的第一个位置	s.indexOf("ef") 的值为 4
s.lastIndexOf(ch)	求字符串 s 中从右向左字符 ch 出现的第一个位置	s.lastIndexOf('c') 的值为 2
s.toCharArray( )	将字符串 s 转换为字符数组	s.toCharArray( ) 的值为 "abcdefGH"
s.trim( )	删除字符串 s 的前后空格	s.trim( ) 的值为 "abcdefGH"
s.replace(ch1,ch2)	将字符串 s 中字符 ch1 替换为 ch2	s.replace('c','X') 的值为 "abXdefGH"
s.equals(s1)	判断字符串 s 与字符串 s1 是否相等	s.equals("abcdefgh") 的值为 false
s.compareTo(s1)	判断字符串 s 与字符串 s1 的大小。若相等，返回 0；若 s > s1，返回正整数；若 s<s1，返回负整数	s.compareTo("abcdefGH") 的值为 0

任务3 运动场馆的人员疏散问题。

公共场所的建设必须考虑到人员的安全疏散问题。在设计奥林匹克运动场馆时，经测算，每位观众走出大门的时间约为3到5秒，同时走出同一扇大门的观众最多可达5人。对于一个可容纳1.2万人的运动场馆，要求在半小时内安全疏散全部观众，那么该场馆至少应该设计多少个疏散出口呢？请你编写程序求解。

①算法分析

这个问题首先要以一扇门为标准思考。总人数为1.2万人，每次从门口出去5个人，则走过大门最多需要12 000/5次。

每次通过大门的时间为3到5秒，则可以用3至5之间的随机数来表示。产生(m，n)之间的随机数的表达式为（可以带小数）：

```
(n-m)*Math.random( )+m;
```

将1至12 000/5次走出大门的循环过程中产生的随机数进行累加，得到12 000/5次经过一个大门时用的总时间。

将求得的总时间除以安全疏散全部观众规定的时间，即可以得出该运动场馆应设计几扇大门。

②程序实现

```
public class chukou
{
   public static void main(String args[ ])
   {
       double t;                    // 声明变量 t，用以存放每人通过门的时间
       double n=12000/5;            // 声明变量 n
       double tt=30*60;             // 声明变量 tt，用以存放规定的时间（半小时）
       double s=0;                  // 声明变量 s，用以存放总时间
       for( int i=1; i<n+1; i++)
       {
           t = (5-3)*Math.random( ) + 3;   // 产生 3~5 之间的随机数
           s = s + t;                      // 累计时间
       }
       int doors=(int) (s / tt);           // 计算需要的大门数
       System.out.println(" 应设计的大门数是："+doors);    // 输出结果
   }
}
```

1．分析程序代码并写出下面程序段的运行结果。

```
public class Exdo
{
    public static void main(String args[ ])
    {
      int i = 1, sum = 0;
      do
      {
        if ((i % 3) == 0) sum += i;
        i++;
      }while (i < 10);
      System.out.println("sum=" + sum);
    }
}
```

2．下面程序的运行结果为：

```
1 2 3 4 5
```

试把程序中 for 循环分别用 while 与 do…while 循环替代，改写程序。

```
public class Ex2_4_2
  {
    public static void main(String args[ ] )
    {
        for (int i=1;i<=5;i++)          // 声明并初始化循环变量 i，设置循环终值 i<=5
        {   System.out.print(i+" ");
        // 执行循环体，输出 i 的值
        }
        System.out.println( );
    }
}
```

3．计算数列 $1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\cdots+\frac{1}{n}$，加到多少项时，其和可超过 10。

4．试分别用三种循环语句编写程序，求出 1 ~ 100 之间能被 3 和 7 同时整除的整数。

第五节 方法与模块化程序设计

程序模块化是结构化程序设计的核心。本节通过实例学习利用模块化程序设计思想来解决实际问题。

一个复杂的问题，可以分解成若干个具有独立任务的子问题来解决。在程序设计中，这些相对独立的任务常常被设计成相对独立的程序段，专业人员称之为模块，并且为每个模块起一个名字，需要时就调用它。这种“分而治之”的处理方法叫做模块化程序设计。

程序设计中往往按功能划分模块，每个模块都必须容易理解和实现，一个功能模块一般仅解决某一个具体问题。各个模块之间通过参数进行联系。

在 Java 中，这些能执行特定任务的模块统称为方法（Method）。方法由一系列具有特定功能的可执行语句组成，这些语句以及相关的数据被封装在一起。在其他程序设计语言中，方法也被称为函数、过程或子程序。我们可以利用方法实现模块化的程序设计。

一 方法的定义与调用

Java 应用程序至少要包含一个 main( ) 方法。main( ) 方法是 Java 应用程序的入口。程序从 main( ) 方法开始执行。按照编程习惯，一般不在 main( ) 方法中编写很多代码，而是把这些代码按照功能编写成方法，放在 main( ) 方法的外部，然后在 main( ) 方法中调用它们。

1. 方法的定义

方法的一般格式如下：

```
[修饰符] 返回类型 方法名(参数序列)
    {
        方法体
    }
```

例如，建立一个计算 n^3 的方法。

首先给方法起一个名字：cub

```
static int cub(int n)        // 声明方法 cub( ) 是静态的，返回值为 int 型
  {
    int c;
    c= n*n*n;
    return c;
  }
```

说明：

（1）修饰符

[private / public] [static]。

（2）修饰符说明

修饰符 public 修饰的方法可以被任何类中的方法访问；

修饰符 private 修饰的方法只能被本类中的其他方法访问；

修饰符 static 修饰的方法叫静态方法或类方法。

（3）方法名

方法的命名应遵循变量命名的约定。参数可以声明为 Java 的任意数据类型，多个参数之间用逗号隔开。如：static max(int n1，int n2，int n3)。

（4）返回类型

返回类型为方法返回值的数据类型，如果没有返回值，则用 void 来修饰方法名，前面章节学习和使用过的 public static void main(String args[ ])，没有返回值，就用 void 修饰。

若方法有返回值，则在方法体中用 return 语句把方法的处理结果返回到调用程序。return 语句用来返回“表达式”的值，一个方法最多返回一个值。如果不需要返回值则不需要使用 return 语句。

2. 方法的调用

在 Java 中，方法建立之后就可以调用了。由于要直接调用 cub( ) 方法，所以必须用关键字 static 对其进行修饰，否则编译会出错，如下面的出错信息：

```
Non-static method cub(int) cannot be referenced from a static context
```

例如，计算 1 到 5 的立方程序中调用静态方法（类方法）cub( ) 的过程。

```
for (int i=1;i<=5;i++)                    // i 依次从 1 到 5
  {
    System.out.println(cub(i));           // 调用静态方法 cub（ ），输出结果
  }
```

代码中，for 循环调用了 5 次 System.out.println( ) 方法。在 System.out.println( ) 方法中，又调用 cub( ) 方法，把 i 的值作为参数 n 传递给 cub( ) 方法。例如，在第 3 次循环中，i=3，所以在 cub( ) 方法中的参数 n 初始化为 3，然后由表达式 n*n*n 计算出值为 27，并把值返回给 System.out.println( ) 方法，System.out.println( ) 方法把该值打印出来。

方法与调用该方法的程序代码交换信息是通过参数传递的。它们的参数名不必相同，但参数的个数、类型必须相同。

在上面的程序中，调用方法 cub( ) 时，传递参数的过程如图 2-5-1 所示。

图2-5-1　调用cub( )方法的参数传递过程

方法 System.out.println( ) 在调用 cub( ) 方法的过程中，参数 i 已经赋值，执行调用后，把参数 i 的值传递给 cub( ) 方法的参数 n。当 cub( ) 方法执行后，将返回值再传递给 System.out.println( ) 方法。把调用方法 cub( ) 的程序代码放到 main( ) 方法中，就成为一个完整的程序，其代码如下：

```
public class Cub2
{
   public static void main(String args[ ])
   {
      for (int i=1;i<=5;i++)                    // i 依次从 1 到 5
      {
         System.out.println(i+" "+cub(i));   // 输出结果
      }
   }
   // 定义方法 cub( )
   static int cub(int n)
   {
         int c;              // 声明变量 c，存放运算结果
            c=n*n*n;         // 求出 n 的立方
         return c;           // 返回结果
    }
}
```

上述程序的 cub( ) 方法中的变量 c 是局部变量，它的作用范围仅限于 cub( ) 方法内部。在方法内部声明的变量叫做局部变量。局部变量必须进行初始化，并且只能在本方法内使用。

例如，在下面的程序中，方法 max(x,y) 定义了两个参数 x 和 y，其功能是求出两个数的最大值。

```
public class Max2
  {
     public static void main(String args[ ])
     {
        int a,b;                                  // 声明变量 a，b
        // 如果输入的参数不是 2 个，那么提示重新输入
        if(args.length!=2)
        {
        System.out.println("Please Input 2 numbers!");        // 提示输入 2 个参数
        return;                                   // 使用语句 return 退出程序
        }
        a = Integer.parseInt(args[0]);        // 将第 1 个参数转换为 int 型，并赋值给 a
        b = Integer.parseInt(args[1]);        // 将第 2 个参数转换为 int 型，并赋值给 b
        int m=max(a,b);                           // 调用方法 max( )，把结果赋值给 m
        System.out.println("Max="+ m);     // 输出结果
     }
```

```
    // 定义方法 max( )，求两个数的最大值
    static int max(int m,int n)
    {
        if (m>n) return m;                    // 如果 m>n，那么返回最大值 m
        else return n;                        // 否则返回最大值 n
    }
}
```

在程序的 main( ) 方法和 max( ) 方法中，分别有一个同名变量 m。但它们都是局部变量，只在各自的方法中有效，两者没有任何关系。

在上面的程序中，调用方法 max( ) 时，传递参数的过程如图 2-5-2 所示。

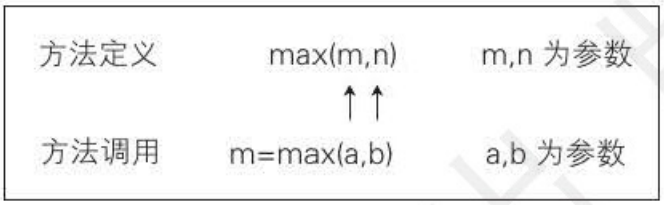


图2-5-2　调用max( )方法的参数传递过程

3. 方法的重载

Java 允许方法重载（Overload）。方法的重载指在一个类中有两个以上的方法，方法名相同，但使用的参数类型或者参数个数不同。

对比 Max2 和 Max3 两个程序。程序 Max3 中，包含了两个同名的 max( ) 方法，增加的这个 max( ) 方法包含三个参数，它的功能是调用包含两个参数的 max( ) 方法，实现从三个数中求出最大值。由于声明的两个 max( ) 方法的参数不同，所以称为 max( ) 方法的重载。程序 Max3 使用了方法的重载。

```
// 方法重载
public class Max3
{
   public static void main(String args[ ])
   {
      int a,b,c;                              // 声明变量 a，b，c
      // 如果输入的参数不是 3 个，那么提示重新输入
      if( args.length != 3 )
      {
         System.out.println("Please Input 3 numbers!");  // 提示输入 3 个参数
         return;                              // 使用语句 return 退出程序
      }
      a = Integer.parseInt(args[0]);          // 将第 1 个参数转换为 int 型，并赋值给 a
      b = Integer.parseInt(args[1]);          // 将第 2 个参数转换为 int 型，并赋值给 b
      c = Integer.parseInt(args[2]);          // 将第 3 个参数转换为 int 型，并赋值给 c
      int m=max(a,b,c);                       // 通过方法重载，调用方法 max( )
```

```
        System.out.println("Max="+ m); // 输出结果
    }
    // 定义方法 max(int m,int n)，从 m，n 中求出最大值
    static int max(int m,int n)
    {
        if (m>n) return m;
        else     return n;
    }
    // 定义方法 max(int n1,int n2,int n3)
    static int max(int n1,int n2,int n3)
    {
        return max(max(n1,n2),n3);     // 调用方法 max( )，从 3 个数中求出最大值
    }
}
```

在上面的程序中，调用方法 max( ) 时，传递参数的过程如图 2-5-3 所示。

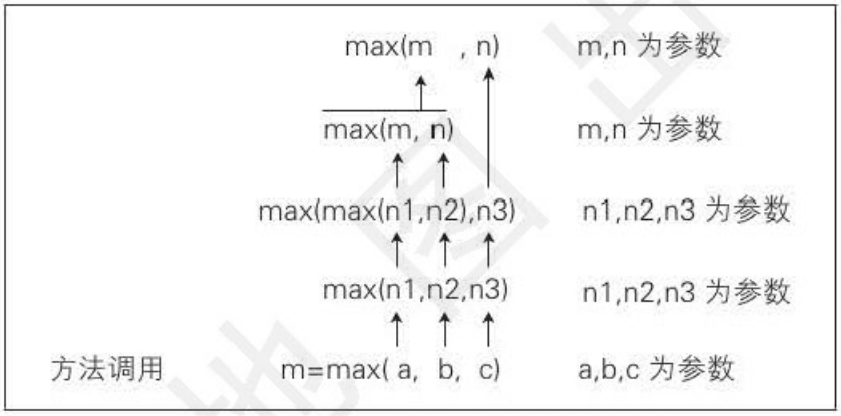


图2-5-3 方法重载时的参数传递过程

二 应用实践

任务1 为选手打分。

为了丰富校园文化艺术生活，学校要举办一场校园歌手大奖赛。有 10 名裁判为参赛选手打分，分值在 1 ~ 100 分之间。评分规则是：去掉一个最高分和一个最低分，其余 8 个得分的平均值为选手最后得分。请编写程序实现为每位选手打分。

①算法分析

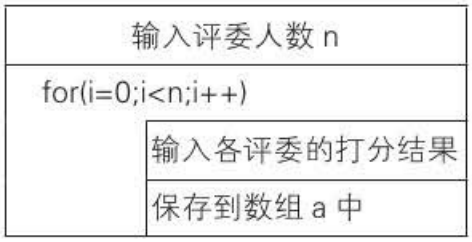
在解决一个较复杂的问题时，可以根据问题的条件和要求，先粗略地划分解决问题的步骤，即用抽象的算法进行描述，然后再针对抽象出来的每一个步骤设计算法，这种设计算法的方法称为自顶向下的设计方法。

◆自顶向下设计算法

流程图如图 2-5-4 所示。

◆逐步求精

模块 1：输入数据



模块 2：累加求总分

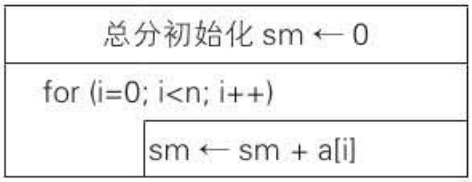
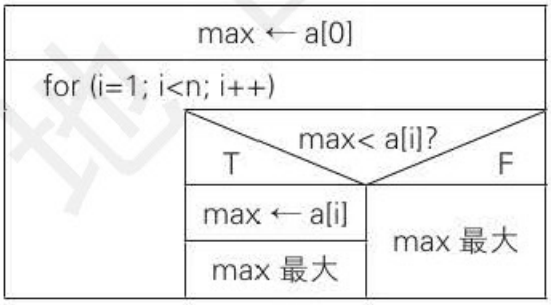


图2-5-4　自顶向下设计算法的流程图

模块 3：求最高分

从一个数组 a 中找出数组元素的最大值，可以利用循环来实现。先把 a[0] 中的数据放到一个用以存放最大数的变量 max 中，然后再逐一进行比较。先把 a[1] 与 max 加以比较，如果 a[1] 大于 max，则用 a[1] 中的数据取代 max 中已有的数；再把 a[2] 与 max 加以比较，如果 a[2] 大于 max，则用 a[2] 中的数据取代 max 中已有的数。如此，依次把数组中的其他数据与 max 进行比较，每次比较后把较大数放入 max 中，最后 max 中的值就是数组元素中的最大值。



模块 4：求最低分

用变量 min 存放最低分，求最低分的框图与模块 3 相同（请自己绘出）。

②编程实现

完整的程序代码如下：

```
// 校园歌手大奖赛评分程序
import java.io.*;
public class Pingfen
{
    static int n;              // 声明变量 n，保存评委人数
    static double a[ ];        // 声明数组 a，保存评分
    static void inputArray( ) throws IOException
```

```
{
    String s;
    String ss[ ];
    // 使用缓冲区 (BufferedReader) 从文本数据流读取文本数据
    InputStreamReader reader=new InputStreamReader(System.in);
    BufferedReader input=new BufferedReader(reader);
    System.out.println(" 输入评委人数 :");       // 提示输入信息
    s=input.readLine( );                          // 从键盘输入一个字符串
    n=Integer.parseInt(s);                  // 将字符串 s 转换为整数 ( 评委人数 )
    ss=new String[n];                       // 初始化字符串数组对象 ss
    a=new double[n];                        // 初始化数组对象 a
    for(int i = 0; i < n; i++)
    {
        System.out.println((i+1)+"> 输入评分 :");    // 提示输入信息
        ss[i]=input.readLine( );                      // 从键盘输入一个字符串
        double x=Double.parseDouble(ss[i]);           // 将字符串 ss 转换为浮点数
        a[i]=x;                                 // 然后保存到数组 a 中
    }
}
static void outputArray(double a[ ])            // 输出数组
{
    for(int i = 0; i < a.length; i++)
    {
        System.out.print(a[i]+" ");
    }
}
static double sum(double a[ ])                  // 评分累加
{
    double sm=0;
    for(int i = 0; i < a.length; i++)
    {
        sm=sm+a[i];                             // 累加
    }
    return sm;
}
static double maxOfArray(double a[ ])     // 求最高分
{
    double max = a[0];                          // 先把数组元素 a[0] 作为最大值
    // 将 max 逐一与数组其他元素比较，把最大值存入 max
```

```
        for(int i = 1; i < a.length; i++)
        {
           if (max<a[i]) max=a[i];
        }
        return max;                                  // 返回最高分
    }
    static double minOfArray(double a[ ])     // 求最低分
    {
        double min = a[0];                           // 先把数组元素 a[0] 作为最小值
        // 将 min 逐一与数组其他元素比较，把最小值存入 min
        for(int i = 1; i < a.length; i++)
        {
           if (min>a[i]) min=a[i];
        }
        return min;                                  // 返回最低分
      }
      public static void main(String args[ ]) throws IOException
      {
        inputArray( );                          // 调用方法 inputArray( ) 输入数据
        outputArray(a);                         // 调用方法 outputArray( ) 输出评分情况
        double sm=sum(a);                       // 调用方法 sum( ) 计算总分
        double mx=maxOfArray(a);                // 调用方法 maxOfArray( ) 计算最高分
        double mn=minOfArray(a);                // 调用方法 minOfArray( ) 计算最低分
        double fen=(sm-mx-mn)/(n-2);            // 去掉最高分和最低分后求平均分
        fen=(Math.floor(fen*100+0.5))/100;      // 保留两位小数计算得分
        System.out.println(" 该选手的最后得分 : "+fen);
      }
}
```

小资料　从面向过程的程序设计到面向对象的程序设计

20 世纪 60 年代，随着计算机应用的逐渐普及，对规模庞大的应用软件的需求逐渐增多。因此计算机科学家提出了“软件工程”的概念，即采用工程的原理、技术和方法来开发和维护软件，从而逐步完善“自顶而下、逐步求精”的结构化模块设计思想。这种程序设计思想仍是围绕过程，通过一系列指令序列来处理数据：程序什么时候开始执行，什么时候输入数据，什么时候调用过程，什么时候输出数据，什么时候结束运行，都通过指令来完成。这就是“面向过程”的程序设计思想。它的特点是由过程来控制数据，过程与数据分离。像 Fortran，C，Basic，Pascal 等语言，都采用这种面向过程的程序设计方

法，并有效地解决了许多实际问题。

到了20世纪80年代，随着计算机技术的飞速发展，面向对象的程序设计方法出现了。Windows下的程序设计语言，如Visual C++和Delphi等都是面向对象的程序设计语言。

面向对象的程序设计在编程思想上同传统的软件开发不同，它需要开发人员转变传统开发中所具有的惯性思维方式。它的根本目的是使程序员更好地理解和管理庞大而复杂的程序。它能够在结构化程序设计的基础上完成进一步的抽象，这种在设计方法上更高层次的抽象正好适应软件开发的特点。

面向对象的程序设计中最基本的概念是对象，一般意义上的对象指一个类的实例化实体，其中包括了特定的数据和对这些数据进行的操作。对象的核心概念就是通常所说的“封装性”“继承性”和“多态性”。

使用面向对象的程序设计方法绝不是要摒弃结构化程序设计方法，它在充分吸收结构化程序设计优点的基础上，引进了一些新的、强有力的概念，从而开创了程序设计工作的新天地。面向对象的程序设计方法把可重复使用性视为软件开发的中心问题，通过装配可重复使用的部件来生产软件，而不是像通过调用函数库中的函数那样来实现。事实上，在对象内部的实现上，我们常常使用过程式的结构化程序设计方法，然而从程序的总体结构上说，它是由一系列对象构成的，对象之间能够以某种方式进行通信和协作，从而实现程序的具体功能。

Java语言是20世纪90年代产生的一种程序设计语言，它简单、面向对象、不依赖于机器的结构，具有重用性、扩展性、可移植性、可控制性、安全性等特点，大大提高了编程效率和程序自动化水平。最大限度地利用网络也是Java语言的一大特点，Java的Applet小程序可在网络上传输而不受CPU和环境的限制。

目前，典型的、完全面向对象的程序设计语言的代表是C++和Java。Java的出现给计算机行业吹来了一股清风，它带来了很多新鲜而有趣的思想和观念，它甚至改变了人们使用计算机的方式，就连WWW的创始人也说：“计算机行业发展的下一个浪潮就是Java，并且很快就会发生。”

1．分析下列程序，写出运行结果。

程序一：

```
public class Lx_1
{
    static void s1( )
    {
        System.out.println("##");
        s2( );
        System.out.println("%%");
    }
    static void s2( )
    {
        System.out.println("$$$$");
```

```
    }
    public static void main(String args[ ])
    {
        System.out.println("*");
        s1( );
        System.out.println("@");
    }
}
```

程序二：

```
public class M_3
{
    int m(int n)                              // 建立带返回值的方法
    {
        int k=0;
        for(int num=1;num<n;num++)
        {
            System.out.println(num+" "+num*num);
            k=k+1;
        }
        return k;
    }
    public static void main(String args[ ])
    {
        M_3 m3 = new M_3( );
        int number=m3.m(6);
        System.out.println("number= "+number);
    }
}
```

2. 在数学中，运算符“!”叫做阶乘，n!=1×2×3×…×n。试将阶乘的计算写成一个方法，并编写一个测试程序计算 10！。

第六节 面向对象的程序设计

在面向对象的程序设计中，一切事物都是对象，每个对象都属于某种类型。本节学习面向对象的基本知识（类的定义、类的封装与继承），通过学习，我们将领悟面向对象程序设计的基本思想，体验面向对象设计程序的优越性。

早在 20 世纪六七十年代，计算机科学家们就提出并完善了“自顶向下、逐步求精”的结构化模块设计思想，即面向过程的程序设计思想。这种思想围绕过程，通过一系列指令序列来处理数据，过程与数据分离。例如 QBasic，C 和 Pascal 等语言，都采用这种程序设计方法。

随着计算机技术的发展，出现了面向对象的程序设计（Object Oriented Programming，OOP）方法。在这种思想指导下设计的程序，总体结构上，是由一系列类和对象 (Object) 构成的，不过，对象内部常常使用面向过程的结构化程序设计方法来实现。例如 C++ 和 Java 等都是面向对象的程序设计语言。

一 类与对象

在 Java 的世界里，一切都是对象。不同的对象通常具有一些共同的特性，把这些共性抽象出来就形成类（Class）。例如，所有的汽车都有车轮、方向盘、刹车板、发动机等装置，把这些属性抽象出来，就形成了汽车类。某一辆具体的汽车，就是汽车类的一个实例（Instance），即对象。

1. 类的声明

在 Java 中，类是组成程序的基本要素。Java 定义了许多类供程序设计使用，也允许我们设计自己的类。

编写 Java 应用程序其实就是编写一个或多个类。这些类中只能有一个主类，用 public 来声明，这个主类必须有一个 main( ) 方法。

类通过关键字 class 来声明，一般格式为：

```
[ 类修饰符 ] class 类名
{
    类体
}
```

（1）类修饰符

Java 语言中主要有三个类修饰符，其功能如表 2-6-1 所示。

表2-6-1 类修饰符

类修饰符	功 能
public	声明此类为公共类，可以被所有的类使用
abstract	声明此类为抽象类，作用是让其他的类继承它的特性
final	声明此类为终结类，不能再被其他的类继承

一个类，如果前面没有 public 和其他修饰符，但是只要它与另一个类在同一个文件或文件夹中，也可以被另一个类访问。

（2）类体

类体是类声明中花括号所包括的程序。类提供的所有变量和方法都要在类体中定义，类体的一般格式为：

```
{
    [修饰符] 变量类型  变量名
    ……
    [修饰符] 变量类型  变量名
    [修饰符] 返回类型  方法名（参数序列）
    {
        方法体
    }
    ……
    [修饰符] 返回类型  方法名（参数序列）
    {
        方法体
    }
}
```

（3）成员变量

在类体中定义的变量叫做成员变量。类中可以声明一个或多个成员变量，也可以没有成员变量。成员变量的类型可以是 Java 中的任意基本数据类型，也可以是某个类。成员变量前面的修饰符，主要用于设置变量的使用方式，如表 2-6-2 所示。

表2-6-2 几个主要的成员变量修饰符

修饰符	功 能
public	声明此变量可被任何类访问（公共的）
private	声明此变量只在本类中可以访问（私有的）
friendly（缺省）	声明此变量可被本包中的所有类访问
protected	声明此变量可被其子类和本包中的其他类访问（受保护）
private protected	声明此变量在本类及其子类中可以访问
static	声明此变量存在于类中而不是对象中，也叫静态成员变量（或类变量）
final	声明此变量的值不能被改变，相当于常量

（4）方法

在类中定义的方法是完成某种功能的程序块，表明如何对数据进行操作。方法的返回类型比变量类型多一个 void 类型，表示不返回任何值。方法名前的修饰符主要用于设置方法的适用范围及其使用方式，如表 2-6-3 所示。

表2-6-3 几个主要的方法修饰符

修饰符	功 能
public	声明此方法可以被任何类中的方法访问（公共的）
private	声明此方法只能被本类中的方法访问（私有的）
friendly（缺省）	声明此方法可以被本类以及本包中的其他类方法访问
protected	声明此方法可以被其子类和本包中的其他类方法访问
static	声明此方法为静态方法（也叫类方法）
final	声明此方法不能被子类重载
abstract	声明此方法为抽象方法，必须在子类中重载

上表中，类方法从属于整个类，而不是从属于某个类的实例。所以在没有建立类的实例的情况下，类方法就可以使用。

例如，设计一个计算圆面积的类。

```
public class Circle
{
    public static final double PI = 3.14;     // 声明常量 π
    public static double r;                   // 声明半径变量
    public double cArea( )                    // 声明计算圆面积方法
    {
        return PI*r*r;
    }
}
```

在 Circle 类中，class 前的修饰符 public 声明 Circle 类可以被其他类访问。

成员变量 PI 和 r 分别表示圆周率和圆的半径。变量 PI 前面用了 public，final，static 三个修饰符来声明变量 PI 是公共的、静态的，而且是一个常量。静态变量表示该类的所有对象都共享同一个静态成员变量。

方法 cArea( ) 的功能是计算圆面积。它的返回值是圆的面积。

Java 程序设计中的主要工作就是定义类、创建类的对象和编写操作数据的方法。

在第五节中，我们已经学习了静态方法调用：方法名（参数），例如求两个数最大值的 max( ) 是静态方法，可通过 m=max(a,b) 来直接调用。

本节程序中方法 cArea( ) 没有用 static 修饰，即这个方法是非静态方法，非静态方法的调用必须先创建对象然后才可以访问。

2. 对象的创建

访问一个类的成员时，通常需要先创建该类的对象，创建对象的过程即类的实例化过程，一个类可以创建多个实例。

在 Java 中，使用关键字 new 来创建类的对象，例如创建 Circle 类的对象 c，代码如下：

```
Circle  c=new  Circle( );
```

类的对象创建后，通过使用“.”运算符实现对类的成员变量和方法的访问。

格式为：对象名 . 方法名 (参数)。

例如，s=c.cArea( )；表明调用对象 c 的 cArea( ) 方法。

现在我们要在 Circle 类中再建立一个 main( ) 方法，在 main( ) 方法内，创建 Circle 类的对象 c，并调用方法 cArea( ) 求出圆的面积 s。完整的程序代码如下：

```
// Circle 类
public class Circle
{
    public final static double PI=3.14;            // 声明成员变量 PI
    public static double r=10;                     // 声明成员变量 r
    public double cArea( )                          // 声明方法 cArea( )
    {
        return PI*r*r;                             // 返回运算结果
    }
    public static void main(String args[ ])        // 应用程序主方法
    {
        Circle c=new Circle( );                    // 创建 Circle 类的对象 c
        double s;                                  // 声明局部变量 s
        s=c.cArea( );                              // 访问对象 c 的 cArea( ) 方法
        System.out.println("s= "+s);               // 输出结果
    }
}
```

构造方法

构造方法（Constructor）是类的一种特殊方法。在 Java 中，它的名字必须与类名完全相同，并且没有返回值，即 void 类型（void 可以省略不写）。

每个类都有一个或多个构造方法，常用来对类的成员进行初始化。如果在定义类时没有定义构造方法，那么 Java 会自动建立一个无内容、无参数的构造方法。

例如，为机器人的类 Robot 赋初值的程序代码如下：

```
class Robot
```

```
{
    String status;
    int speed;
    int power;
    Robot(String a1, int a2, int a3)
    {
        status = a1;
        speed = a2;
        power = a3;
    }
}
```

在主方法中，可以用关键字 new 创建 Robot 类的一个实例，如：

```
Robot  Rb =new  Robot("exploring", 8, 200)
```

其中，将 Robot 类的对象 Rb 的属性 status 设置为“exploring”，将 speed 属性设置为 8，将 power 属性设置为 200。

二 面向对象程序设计的基本特征

面向对象的程序设计有三个基本特征：封装（Encapsulation）、继承（Inheritance）和多态（Polymorphism）。

1. 封装

简单地说，封装就是把一段程序包装起来，只向用户提供一些变量和方法，而隐藏其内部实现的细节。类本身就是一种封装。前面计算圆面积的类就是将变量 r 和方法 cArea( ) 封装在 Circle 类中的例子。

2. 继承

在面向对象的程序设计中，可以充分利用原有的程序代码，这一特征就是继承。它体现了类与类之间的一般与特殊的关系。被继承的类叫做父类或超类（Super Class），通过继承来实现的类叫做子类（Subclass）。通过继承，子类可以重用父类的代码，同时也允许子类添加自己特有的属性和操作。

Java 中通过关键字 extends 来实现继承。继承在声明子类时说明，例如以下代码定义了 Applet 类的子类 HelloApplet。

```
public class HelloApplet extends Applet
{
    类体
}
```

子类中对父类成员变量和方法的访问是有条件的，即子类只能访问在父类中通过 public 或 protected 修饰的或默认友好的（无修饰词或修饰词为 friendly）成员变量和方法。

我们以图形类“形状”为例（图 2-6-1），来说明如何实现类的继承。

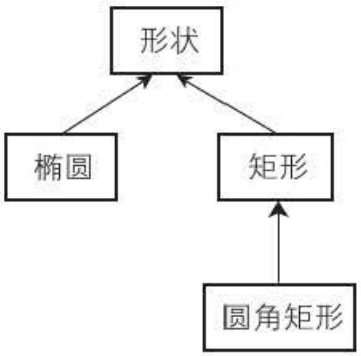


图2-6-1　类的继承示意图

为了突出继承关系，这里画图语句暂用输出相应文字的语句代替，待我们掌握了绘制图形的类及其方法之后，再将程序代码中 draw( ) 方法的绘图功能加以实现。

程序代码如下：

```
// 源文件名为：Inherit.java
// 先设计父类 Shape（形状）
class Shape                              // 父类 Shape 类的声明
{
    protected int lineSize;              // 声明成员变量 lineSize，表示线宽
    public Shape( )                      // 声明方法 Shape ( )
    {
      lineSize = 1;
    }
    public void setLineSize(int ls)      // 设置线宽，声明方法 setLineSize( )
    {
      lineSize = ls;
    }
    public int getLineSize( )            // 获得线宽，声明方法 getLineSize( )
    {
      return lineSize;
    }
    public void draw( )                  // 声明绘图方法 draw( )，以输出文字示意
    {
      System.out.println("draw a shape");
    }
}
// 再设计子类 Rectangle（矩形）
class Rectangle extends Shape            // 声明子类 Rectangle，继承父类 Shape
{
```

```
    int left;              // 成员变量 left( 矩形左上角 X 坐标 )
    int top;               // 声明成员变量 top( 矩形左上角 Y 坐标 )
    int width;             // 声明成员变量 width ( 矩形宽度 )
    int height;            // 声明成员变量 height ( 矩形高度 )
    public Rectangle(int l, int t, int w, int h)          // 声明方法 Rectangle ( )
    {
      super( );          // super( ) 表示调用超类中的构造方法
      left = l;
      top = t;
      width = w;
      height = h;
    }
    public void draw( )                    // 声明绘图方法 draw( )，以输出文字示意
    {
      System.out.println("draw a rectangle");
    }
}
// 再设计子类 RoundRect ( 带圆角的矩形 )
class RoundRect extends Rectangle      // 声明子类 RoundRect，继承父类 Rectangle
{
    int radius;                            // 声明成员变量 radius ( 圆角半径 )
    public RoundRect(int l, int t, int w, int h, int r)
    {
      super(l, t, w, h);                   // super( ) 表示调用超类中的构造方法
      radius = r;
    }
    public void draw( )                    // 声明方法 draw( )，以输出文字示意
    {
      System.out.println("draw a round-rectangle ");
    }
}
public class Inherit
{
    public static void main(String args[ ])
    {
      // 创建子类 RoundRect 对象，并调用父类方法设置 lineSize 属性
      RoundRect roundrect = new RoundRect(0, 0, 50, 30, 5);
```

```
        roundrect.lineSize = 4;                    // 访问父类成员变量，设置线宽
        System.out.println("LineSize of round-rect : "+ roundrect.getLineSize( ));
roundrect.draw( );                          // 调用 draw( ) 方法
    }
}
```

上面例子中，子类继承了父类所有的成员变量和方法。

为简化代码的编写，可通过关键字 super 实现对父类成员的访问，并在子类的同名方法中添加所需要的内容。程序中用 super( ) 表示父类中的同名方法，如果子类中有 draw( ) 方法，就执行子类中的 draw( ) 方法，如果没有 draw( ) 方法，就执行父类中的同名方法。

在三个类中，都有一个同名的方法 draw( )，当子类 RoundRect 在调用 draw( ) 方法时，究竟是调用子类的方法还是调用父类的方法呢？我们可以做一个小实验，先将子类 RoundRect 中的 draw( ) 方法用“//”屏蔽掉，看运行结果；然后再将子类 Rectangle 中的 draw( ) 方法也屏蔽掉，看运行结果。通过比较即可发现问题。请把你的理解和体会写下来存入电子学习档案袋“我的感受”中，并以“定稿”方式发布。

3. 多态

当某个子类继承父类后，可能需要修改父类中某些方法的实现方式，即在子类中重写父类中已有的方法，也可能在一个类中有多个具有相同方法名而参数个数、参数类型不同的方法。当方法被不同的对象调用时，能产生不同的行为，这种现象称为多态。

面向对象的程序中多态的情况有多种，可以通过子类对父类方法的覆盖（方法重写）实现多态，也可以利用重载在同一个类中定义多个同名的不同方法（方法重载）实现多态。

例如，在第五节的程序 Max3 中声明了 int max(intm，intn) 和 int max(int n1，int n2，intn3) 两个方法，语句 int p=max(a，b，c) 将调用第二个方法，语句 int q=max(i，j) 将调用第一个方法。这就是方法重载。

在关于形状的类中，矩形类 Rectangle 和圆角矩形类 RoundRect 等图形类都继承自形状类 Shape。这些类中都有画图方法 draw( )，虽然方法名称相同，但画图方法并不相同。这就是方法重写，子类可以重新实现父类的某些方法，使其具有自己的特征。

接口

假如我们已经创建了椰子类和球形类，椰子类具有水果的特征，球形类具有旋转、投掷等特征。如果你想让椰子是一个球形的对象，在 Java 中如何实现呢？在 Java 语言中不允许多重继承，即让一个类继承多个类。在实际程序设计中也会遇到类似的问题，即仅靠单重继承不能解决更复杂的问题。于是 Java 语言设计了接口（Interface）的概念，使得程序的类层次结构更加合理，更符合解决实际问题的需要。

Java 中把用于完成特定功能的若干属性组织成相对独立的属性集合，这种属性的集合就是接口。定义接口与定义类非常相似，实际上完全可以把接口理解成为一

种特殊的类。它定义若干抽象方法和常量，形成一个属性集，该属性集通常对应某一组功能。

接口的声明格式如下：

```
[public] interface 接口名          // interface 为关键字
{
        变量声明;
        抽象方法声明;
}
```

接口中的所有变量都必须用 static final 来修饰（在声明时可以省略），也就是说接口中的变量都必须修饰为常量；接口中的所有方法也都必须用 public abstract 来修饰，也就是说接口中的方法都必须是抽象方法（声明时可以省略）。

接口的使用，即接口的实现（Implementation），必须在声明类时同时声明。

接口的实现格式如下：

```
[ 修饰符 ] class 类名  [extends 父类名 ]  implements  接口名
{
    类体
}
```

注意，接口中的方法必须声明为抽象方法，实现接口时，子类中必须将所有的抽象方法加以实现；接口中声明方法时可以隐含 public，但实现方法时不可省去。

例如，图形界面中使用的 ActionListener 就是系统定义的接口，它具有监听并处理动作事件的功能，其中包含了一个抽象方法 public void actionPerformed(ActionEvent e)；所有希望能处理动作事件的类都必须具有 ActionListener 接口定义的功能，即必须覆盖（重写）actionPerformed( ) 方法。

定义接口也必须有修饰符，但只有一个修饰符 public，说明它可以被所有的类和接口使用；而没用 public 修饰符的接口则只能被同一个包中的其他类和接口使用。接口也具有继承性，通过 extends 关键字声明它将继承父接口的所有属性和方法。与类的继承不同的是一个接口可以有一个以上的父接口，接口之间用逗号分隔，形成父接口列表。新接口将继承所有父接口中的属性和方法。

下面是一个关于球形类的接口，功能是让水果类的子类——椰子类既具有水果类的特征，又具有球形类的特征。

```
interface jiekou                    // 声明球形类的接口 jiekou
{
    int radius =6;                  // radius 为半径
    String mycolor=" 棕色 ";        // mycolor 为颜色
    void method1( );                // 声明抽象方法 1
```

```
    String method2(String x );          // 声明抽象方法 2
}
// 声明类 Example 作为椰子类，并使用接口 jiekou
public class Example implements jiekou
{
    int n;
    public void method1( )              // 抽象方法 1 的实现
    {
        System.out.println(" 半径为 : "+radius);
    }
    public String method2(String c)     // 抽象方法 2 的实现
    {
        System.out.println(" 颜色为 : "+c);
        return c;
    }
    public static void main(String args[ ])     // 主方法 main( )
    {
        Example y=new Example( );       // 创建椰子类的一个实例 y
        y. method1( );                  // y 继承了球形类的属性
        y. method2( mycolor );          // y 继承了球形类的属性
    }
}
```

程序输出如下：

```
半径为 : 6
颜色为 : 棕色
```

因为椰子是水果类的子类，所以它不能再继承球形类的特征，但是可以通过接口的方式来实现球形类的特征。上例中的椰子既具有水果特征，又利用接口表现了球形类的特征，如半径、颜色等。

三　常用的Java类库

我们所用的计算器，通常都有一些现成的函数功能，例如求平方、三角函数等。Java语言允许编程者建立自己的“函数”库，这就涉及“包（Package）”的概念。

1. 包

包是一组类的集合（严格说是一系列相关类和接口的集合），也是面向对象思想中封装的一种形式。你可以为自己的几个相关的类创建一个包。

包的声明格式为：

```
package 包名；
```

包的声明必须是程序中第一条非注释语句，而且一个源文件只能有一个包声明语句。例如，欲建立一个名为 mypkg 的包，这个包里含有类 abc，程序代码如下：

```
package mypkg；
public class abc
{
    public static int m =5;
}
```

上述包经过编译生成 abc.class 文件，并且将其移到“mypkg”文件夹中。

包的导入格式为：

```
import 包名；
```

例如，创建一个叫“usemypkg”的类，如果要使用 mypkg 包中的 abc 类，需要先将 mypkg 包导入。

```
import mypkg.abc;
public class usemypkg
{
    public static void main(String args[ ])
    {
        abc a=new abc( );        // 创建 abc 类的对象
        System.out.println(" 调用类 abc 的结果是 :"+a.m);
    }
}
```

以上是导入自己建立的包的简单例子。

Java 中包名对应于文件系统中与包同名的子目录，要使用包中的某个类或类中的方法，需要用 import 导入这个类或这个类所在的包。导入类之后就可以使用这个类中的变量和方法了。

2. Java类库

Java 系统提供了许多基础性的和常用的类供编程使用，这些类按功能的不同分装成若干个包，这些包统称为 Java 类库，也叫标准 Java 包。如 java.io 和 java.lang，它们存放了一些基本类，如 System，String 或 Math。

Java 利用包的形式组织管理类，SDK 就是 Java 提供的软件开发包，其中包含的常用的类库见表 2-6-4。

表2-6-4 SDK常用类库

包 名	内容或功能
java.io	包含处理 I/O 文件的类
java.util	包含为任务设置的实用程序类，如随机数发生、定义系统特性和使用与日期日历相关的函数
java.lang	包含一些形成语言核心的类，如 String，Math，Integer 和 Thread
java.awt	包含构成抽象窗口工具包（AWT）的类，这个包用于构建和管理应用程序的图形用户界面
java.applet	包含可执行 Applet 特殊行为的类
java.net	包含执行与网络相关的操作的类和处理接口及统一资源定位符 (URL) 的类

例如，Math 类包含在 java.lang 包中，使用 Math 类中的方法就需要导入 java.lang 包：

```
import java.lang;
```

实际上，java.lang 包会自动导入，而其他包则必须由 import 导入。

四 代码重用

代码重用的特性在 Java 系统里主要表现在以下几方面：

◆通过类的继承和声明类的实例，可以重复使用封装在类中的代码。

◆方法的重写与重载也体现了代码重用，当子类调用一个对象的方法时，Java 会在该对象的类中寻找方法定义；如果没找到，它会调用上一级的同名方法，如父类的方法。方法的继承使得在子类中重复定义和使用方法时无需复制代码。

◆ SDK 提供的标准包和接口以及第三方提供的扩展类库得到了广泛的使用，充分体现了代码重用的优势。

◆ Java 程序代码可以不经过任何修改，就能在支持 Java 的平台上运行。

计算年平均降雨量。

尝试使用 Java 类库中的 javabook 包，编制一个计算年平均降雨量的程序。要求输入一年 12 个月的降雨量，计算年平均降雨量，以及每月降雨量与年平均降雨量的偏差，并且输出结果。

① 问题的提出

对于初学者来说，掌握 Java 标准类库提供的类不容易。我们应该学会获取和使用现成的 Java 类库。因特网上就有许多容易使用的类库，也有为一些应用而提供的类库。javabook 就是这样的类库之一。

②使用 javabook 包

前面我们已经学习了输入输出的问题，但程序中的输入输出处理实现起来都不是很方便。javabook 类库中包括一些图形界面的类，如 MessageBox，InputBox，OutputBox 等，利用这些类以及很少的代码就可以非常方便地实现图形界面的输入输出。

下面我们就学习用这些类进行编程。这些类都位于 javabook 包中，在教科书配套光盘“zy\tools”中可以找到。使用之前应先将 javabook 包复制到自己的工作目录下。

程序代码如下：

```
// 计算年平均降雨量
import javabook.*;        // 导入 javabook 包，javabook 应放在当前工作目录中
public class RainFall
{
   public static void main(String args[ ])
   {
      // 声明并创建 MainWindow 类对象
      MainWindow mainWindow=new MainWindow( );
      // 声明并创建 InputBox 类的对象
      InputBox inputBox=new InputBox(mainWindow);
      // 声明并创建 OutputBox 类的对象
      OutputBox outputBox=new OutputBox(mainWindow);
      mainWindow.show( );          // 使 mainWindow 可见
      outputBox.show( );           // 使 outputBox 可见
      // 声明并创建数组 rainfall，保存每月的降雨量
      double rainfall[ ]=new double[12];
      // 声明并创建字符串数组 monthName
      String monthName[ ]={" 一月 "," 二月 "," 三月 "," 四月 "," 五月 "," 六月 ",
      " 七月 "," 八月 "," 九月 "," 十月 "," 十一月 "," 十二月 "};
      double sum=0.0;              // 声明变量 sum，保存总降雨量
      double average;              // 声明变量 average，保存平均降雨量
      double d;                    // 声明变量 d，保存降雨量偏差
      // 将每月的降雨量存入数组
      for (int i=0;i<12;i++)
      {
          rainfall[i]=inputBox.getDouble(monthName[i]);
          sum=sum+rainfall[i];     // 累加
      }
   average=sum/12;                 // 计算年平均降雨量
   // 在 outputBox 窗口内输出每月的降雨情况
```

```
    for (int i=0; i<12; i++)
    {
        // 在 outputBox 指定位置输出月份
        outputBox.print(Format.rightAlign(3,monthName[i]));
        // 在 outputBox 指定位置输出每月降雨量
        outputBox.print(Format.rightAlign(17,1,rainfall[i]));
        d=Math.abs(rainfall[i]-average);          // 计算降雨量偏差
        // 在 outputBox 指定位置输出每月降雨量偏差
        outputBox.printLine(Format.rightAlign(17,2,d));
      }
    }
}
```

程序运行后将首先显示 mainWindow 和 outputBox 窗口，然后出现 InputBox 对话框，如图 2-6-2 所示，我们可以按照提示输入数据。

图2-6-2　javabook包的输入对话框

全年 12 个月的数据输入完成后的输出结果如图 2-6-3 所示。

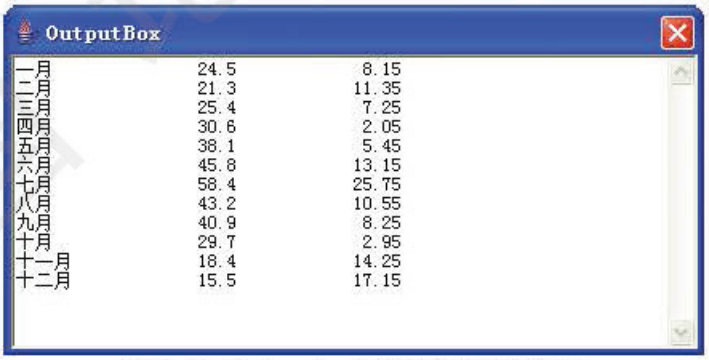


图2-6-3　javabook包的输出信息框

知识拓展

Java 文档

我们可以通过 Java 文档来了解类库的使用。在 Java 的安装目录中，包含了 SDK 以及相关文档 javaDOC。打开 SDK 安装目录“jdk\docs\”中的 index.html 文件，（即 javaDOC 文档的首页），如图 2-6-4 所示。

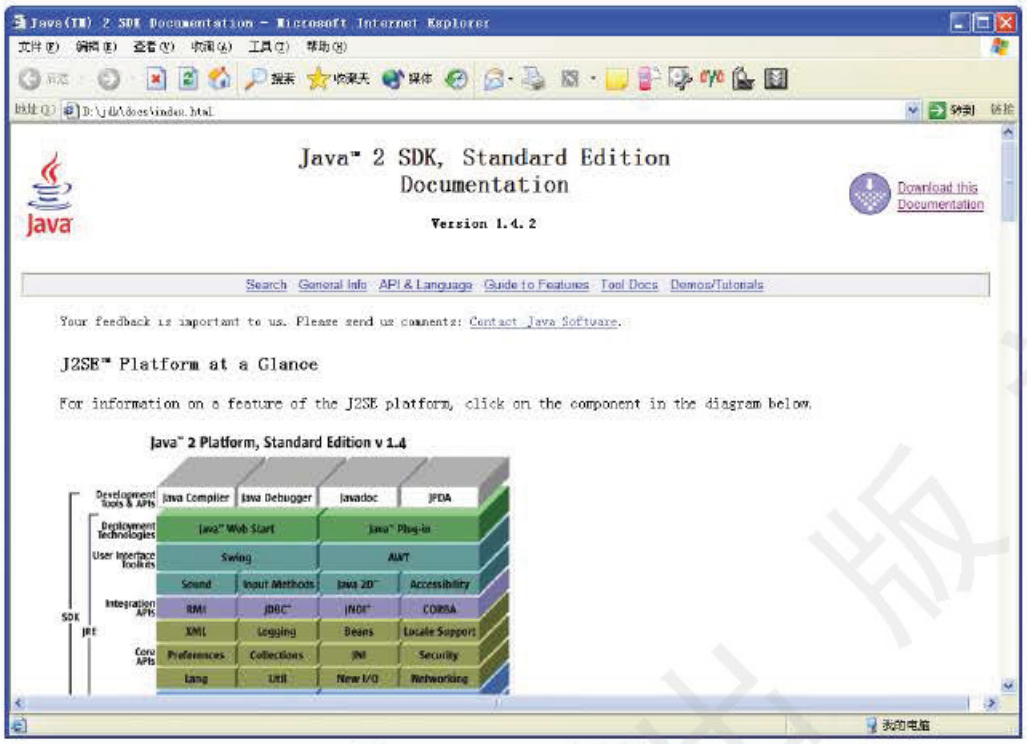


图2-6-4 javaDOC 首页

单击导航条中的“API&Language”或向下拖动窗口右侧的滚动条，单击“Java 2 Platform API Specification”，即可打开Java API文档页面，浏览有关类的使用信息，如图2-6-5所示。

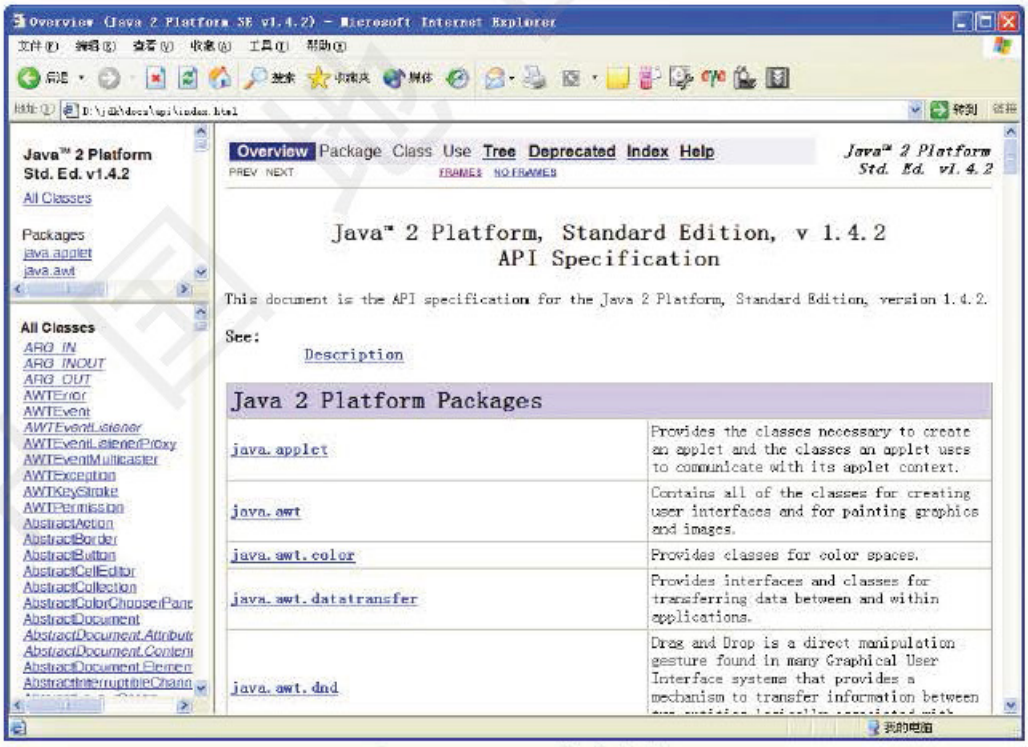


图2-6-5 Java类库文档

1．写出下列程序的运行结果。

```
public class Scope1
{
    int x=1;                                    // 此处变量 x 是类的成员（全局变量）
    public static void main(String args[ ])
    {
        int x=2;                                // 此处变量 x 是局部变量
        System.out.println("x in main= "+x);    // main( ) 方法访问的是局部变量
        Scope1 vt = new Scope1( );              // 先创建对象
        System.out.println("x in class= "+vt.x); // 通过对象访问的是全局变量
    }
}
```

2．从因特网上搜索有关 javabook 类库的文档资料，并使用 javabook 类库中的方法改写第四节奥运会场馆人员疏散程序中输入输出部分的代码。

第七节 图形用户界面的程序设计（选修）

具有友好图形界面的应用程序，能极大地方便用户使用。开发基于图形界面的程序已成为现代程序设计的发展趋势。本节将介绍简单的图形设计、图像处理等程序设计知识，希望大家感兴趣，并深入研究，将来能开发出更多更好的图形界面程序。

基于图形界面的程序设计是程序设计的一个重要方面，它能很好地实现人机交互，用它开发出的应用程序也比较生动、能吸引人。图形界面的设计通常包括图形图像处理、菜单制作、事件响应以及多媒体呈现等。Java 提供的基本类库对图形、线程、事件处理等都提供了很好的支持，特别是利用 Java 的 Applet，能很容易地实现图像显示、音乐播放等多媒体功能。

一 图形与图像的处理

利用 Java 设计图形界面应用程序时，通常需要建立 Windows 窗口，要设计相关的组件（Component），例如按钮、文本框、菜单等，还应该考虑布局（Layout），即怎样在窗口中摆放各个组件。我们把能建立窗口框架的 Frame 或 JFrame 类叫做放置按钮等组件的容器（Container），常用的容器还有 Panel，JPanel，Applet，JApplet 等。

Java 提供的基本图形编程类库 AWT（Abstract Windows Toolkits）以及 Swing 类库，能对图形和动画的程序设计提供很好的支持。其中，Java 的 java.awt.Graphics 类可提供几十个图形设计方法，可以设置颜色、字体等属性，可以完成图形绘制、填充、变换和图像处理等操作。javax.swing 类包含了 java.awt 类的大部分图形方法，已经成为 Java 主要的图形界面开发工具包。

在进行图形设计时，首先要考虑的是在什么位置绘制图形和采用什么样的坐标系统。Java 提供的坐标系统的原点在容器绘图区域的左上角，沿原点水平向右为 x 轴正方向，沿原点垂直向下为 y 轴正方向，即坐标值向右下角增大。绘图区域的左上角坐标总是 (0,0)。Java 绘图方法采用的单位为像素。

1. 绘制图形

在窗口中绘图的一般方法是：先创建窗口，接着在创建窗口的类中增加 paint（）方法，然后在 paint（）方法中编写绘图语句。

（1）创建窗口

建立标准的窗口一般需要三个步骤，如图 2-7-1 所示：

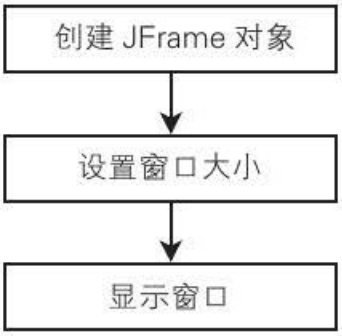


图2-7-1　创建窗口的流程

下面的程序演示了建立窗口的方法。

```
import java.awt.*;                    // 导入 java.awt 包
import javax.swing.*;                 // 导入 javax.swing 包
public class Frame1
{
    public static void main(String args[ ])
    {
        // 创建 JFrame 对象 win1，窗口标题是 Frame1
        JFrame win1 = new JFrame("Frame1");
        win1.setSize(280,180);        // 用 setSize( ) 方法设置窗口大小
        win1.setVisible(true);        // 用 setVisible( ) 方法显示窗口
    }
}
```

程序经编译后运行，结果显示一个标题为“Frame1”的空窗口，如图 2-7-2 所示。

图2-7-2　建立空窗口

（2）添加paint( )方法

在创建窗口的类中增加 paint( ) 方法。其中参数“Graphics　g”声明了一个 Graphics 类型的对象 g。

```
public void paint(Graphics g)
{
}
```

（3）编写绘图语句

Java 提供的 Graphics 类中包含各种绘图方法，可以用来设置颜色、显示文字、绘制图

形等。

◆设置颜色

Graphics 类的 setColor( ) 方法可以设置颜色。java.awt 包中的 Color 类定义了十几种颜色变量，它们都是静态的，可以直接调用。例如：

```
setColor(Color.red);
```

表示设置画图颜色为红色。

◆显示文字

Graphics 类的 drawString( ) 方法可以在窗口中显示文本。格式为：

```
drawString(String str, int x, int y)
```

其中，参数 str 为要显示的字符串，x，y 为字符串第一个字符显示位置的坐标。例如：

```
g.drawString("Hello!",200,80);
```

表示在窗口坐标（200,80）处显示字符串“Hello!”。

如果希望字体有变化，可以先建立 Font 类的对象，然后再调用 Graphics 类的 setFont( ) 方法。

◆绘制图形

Graphics 类的各种绘图方法可以用来在窗口中绘制各种几何图形。例如 drawRect( ) 方法可以用来绘制矩形。格式如下：

```
drawRect(int x, int y, int width, int height);
```

其中 x,y 为矩形左上角的坐标值，width 为矩形宽，height 为矩形高。如果要画出以 (50,50) 为左上角，宽为 200, 高为 100 的矩形，可以设置参数如下：

```
g.drawRect(50,50,200,100);
```

如果要画一个正方形，只要把矩形的高与宽设为等值即可。

下面的程序演示了绘制图形和显示文字的常用方法的使用。

```
import java.awt.*;                          // 导入 java.awt 包
import javax.swing.*;                       // 导入 javax.swing 包
public class Frame2 extends JFrame          // 定义类 Frame2，声明继承了JFrame 类
{
   Frame2(String title)                     // 声明类 Frame2 带参数的构造方法
   {
        super(title);                       // 使用 super( ) 方法创建带标题的窗口
   }
   public void paint(Graphics g)
   {
       g.setColor(Color.blue);              // 设置当前的绘图颜色为蓝色
       // 画以 (50,50) 为左上角，宽为 200、高为 100 的矩形
       g.drawRect(50, 50, 200, 100);
       // 画以 (50,50) 为左上角，长轴为 200、短轴为 100 的椭圆
       g.drawOval(50, 50, 200, 100);
```

```
        g.setColor(Color.red);                    // 设置当前的绘图颜色为红色
        g.drawString("Hello!",200,80);            // 在（200，80）处显示字符串
    }
    public static void main(String args[ ])
    {
        Frame2 win1 = new Frame2("Draw2");  // 创建 Frame2 对象 win1
        win1.setSize(300,200);                    // 设置窗口大小
        win1.setVisible(true);                    // 显示窗口
    }
}
```

程序经编译后运行，结果如图 2-7-3 所示。

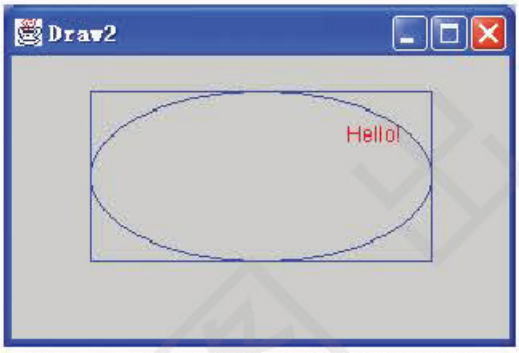


图2-7-3　在窗口中画图与输出文字

编写一个绘制五角星的程序。

绘制五角星的难点在于确定五角星的五个顶点。可以先画一个圆，然后五等分圆的周长，则五个等分点即为五角星的五个顶点。

五个顶点确定后，可以使用多边形绘图方法 drawPolygon( )，按一定顺序将各顶点连接起来，如图 2-7-4 所示。如果你觉得画个五角星很容易，那就充分发挥你的想象力，画出你感兴趣的图形吧。

图2-7-4　在窗口中绘制五角星

知识拓展 Graphics类中常用的绘图方法

java.awt.Graphics 类中提供了几十个绘图方法，常用的方法列在表 2-7-1 中。这些方法的具体使用说明，参见 javaDOC 文档。

表2-7-1 Graphics类中常用的绘图方法

图形	绘制方法	
直线	drawLine(int x1,int y1,int x2,int y2);	// 画直线
矩形	drawRect(int x,int y,int width,int height);	// 画空心矩形
	fillRect(int x,int y,int width,int height);	// 画实心矩形
	drawRoundRect(int x,int y,int width,int height);	// 画圆角空心矩形
	fillRoundRect(int x,int y,int width,int height);	// 画圆角实心矩形
	draw3dRect(int x,int y,int width,int height);	// 画空心立体矩形
	fill3dRect(int x,int y,int width,int height);	// 画实心立体矩形
多边形	drawPolygon(int xPoints[ ],int yPoints[ ],int nPoints);	// 画空心多边形
	fillPolygon(int xPoints[ ],int yPoints[ ],int nPoints);	// 画实心多边形
椭圆	drawOval(int x,int y,int width,int height);	// 画空心椭圆
	fillOval(int x,int y,int width,int height);	// 画实心椭圆
圆弧	drawArc(int x,int y,int width,int height,int startAngle,int arcAngle);	// 画圆弧
	fillArc(int x,int y,int width,int height,int startAngle,int arcAngle);	// 画扇形

2. 显示图像

使用 Graphics 类的 drawImage( ) 方法可以在窗口内显示图像，格式为：

```
Public abstract Boolean drawImage(Image img, int x, int y, ImageObserver obs)
```

其中，参数 img 为 Image 类的对象，x，y 为图像左上角的坐标，obs 为显示图像的地方。

为了显示图像，还需要知道图像文件的地址，因此需要创建图像文件的 URL 对象，URL 类在 java.net 包中。创建 URL 对象时，需要利用“try…catch”结构抛出异常。

下面的程序运行后可以在窗口中显示指定文件的图像：

```
import java.awt.*;                  // 导入 java.awt 包
import javax.swing.*;               // 导入 javax.swing 包
import java.net.*;                  // 导入 java.net 包
public class DisplayImage1 extends JFrame
{
    DisplayImage1(String title)
    {
        super(" 显示图像 ");    // 带参数的构造方法，显示窗口标题
    }
    public void paint(Graphics g)
```

```
    {
        try
        {
            // 创建 URL 对象，设置图像文件地址（路径）
            URL pairURL=new URL("file: 山村水色 .jpg ");
            // 创建 Image 对象 myImage
            Toolkit toolkit = Toolkit.getDefaultToolkit( );
            Image myImage=toolkit.getImage(pairURL);
            // 显示图像。this 指 JFame，作为 ImageObserver 对象
            g.drawImage(myImage,2,10,this);
            // 创建 Font 对象 ,
            Font ft1=new Font(" 隶书 ",Font.BOLD,36);
            g.setFont(ft1);                          // 设置字体
            g.setColor(Color.red);                   // 设置当前的颜色为红色
            g.drawString(" 山村水色 ",200,300);       // 显示字符串
          }
          catch(MalformedURLException e)             // 抛出异常
          {
            g.drawString(" 图形文件未找到！ ",30,30);  // 显示异常信息
          }
    }
    public static void main(String args[ ])
    {
        // 创建 DisplayImage1 对象 Win1
        DisplayImage1 win1 = new DisplayImage1("DisplayImage1");
        win1.setSize(500,360);                 // 设置窗口大小
        win1.setVisible(true);                 // 显示窗口
    }
}
```

程序运行结果如图 2–7–5 所示。

图2–7–5　在窗口中显示图像

二 事件处理

在图形界面应用程序中，单击一个按钮或单击一个菜单项等等，通常就会发送一个消息，程序随即做出反应，从而实现人机交互。我们把类似单击按钮、移动鼠标等这样的动作称为事件，程序的反应称为事件的处理。

在 Java 语言中，事件用类来描述。不同事件对应的类也不同，这些类都属于 EventObject 类的子类。

事件的处理也是由类来完成的。事件的处理就是一个监听事件、解释事件并进行交互处理的过程。在 Java 中，事件的发生和事件的处理是相关的，一般要通过接口来实现。如果发生了 ActionEvent 事件，那么处理这个事件的类应该实现 ActionListener 接口；如果发生了 XXXXEvent 事件，处理该事件的类就要实现 XXXXListener 接口。

事件处理类存放在 java.awt.event 包中。

例如，打开窗口、关闭窗口、改变窗口大小等都属于 WindowEvent 事件，产生事件的对象当然是 JFrame 类的窗口，所以处理这些事件都要通过窗口调用 addWindowListener（）方法来完成。

如果一个接口包含多种方法，那么 Java 可通过一个专门的类来实现接口，这些方法的名字一般为 XXXXAdapter。这样只要继承该类，就可以实现该接口的某个方法。

下面的程序演示了单击窗口关闭按钮事件的处理情况。程序经编译后运行，单击窗口的关闭按钮，可以退出程序，结束运行。

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;                    // 导入事件处理包
public class DrawString1 extends JFrame
{
    DrawString1(String title)
    {
        super(title);
    }
    public void paint(Graphics g)
    {
        g.drawString(" 单击关闭按钮，即可退出窗口！ ",70,100);   // 显示信息
    }
    public static void main(String args[ ])
    {
        DrawString1 win1 = new DrawString1("DrawString1");
        win1.setSize(300,200);              // 设置窗口大小
        win1.setVisible(true);              // 显示窗口
```

```
        }
    }
// 定义继承 WindowAdapter 类的子类 WindowClose
class WindowClose extends WindowAdapter
{    // 单击“关闭”按钮，程序执行 windowClosing( ) 方法
    public void windowClosing(WindowEvent e)
    {
    System.exit(0);                    // 调用 System.exit( ) 方法结束程序运行
    }
}
```

三　播放多媒体

要播放多媒体，首先要知道多媒体文件存放的 URL 地址，然后才能将声音或图像文件打开并运行。使用 Java 的 Applet 小程序可以很好地实现多媒体的播放。

了解 Applet

编写 Applet 小程序时，必须把它的类声明为 public，使它的名称与它所在的文件名相同，还要声明它是 java.applet.Applet 的子类或 javax.swing.Japplet 的子类。因此，Applet 小程序代码中必须有如下语句：

```
import java.applet.*;
```

或

```
import javax.swing.Japplet.*;
```

Applet 小程序的格式如下：

```
import java.applet.*;
public class 类名 extends Applet   // 声明是 Applet 的子类
{
}
```

或

```
import javax.swing.Japplet.*;
public class 类名 extends JApplet    // 声明是 JApplet 的子类
{
}
```

Applet 本质上是图形方式的，因此可以通过创建的 paint( ) 方法直接在 Applet 的面板（Panel）上绘图。paint( ) 方法要求带有一个参数，此参数应是 java.awt.Graphics

类创建的一个实例。利用 paint( ) 方法可以在 Applet 界面上直接完成绘画动作，这样就可以在每次页面刷新时都会重绘界面。

除了 paint( ) 方法，较复杂的 Applet 小程序还会使用到下面的一些方法。

init( ) 方法：用来初始化数据的值。它只在 Applet 首次装入时被调用，并且在调用 start( ) 之前执行完毕。

start( ) 方法：在 init( ) 完成后开始执行。它可以激活 Applet 中的某一行为。

stop( ) 方法：它与 start( ) 的作用相反，可以停止 Applet 中的某一行为。

以前讲到的 JavaApplication 应用程序，都是从 main( ) 主方法进入开始执行的。在 Applet 小程序中，浏览器首先调用 init( ) 方法对 Applet 进行基本的初始化操作，然后再调用 start( ) 方法开始程序的执行。

1. 显示图像

要显示图像，就需要知道图像的 URL 地址。java.net.URL 类可实现文件与 Applet 之间的连接。在 Applet 类中有两个方法可获取 URL 地址。

方法 1：getDocumentBase( );

它返回当前浏览器中带有 Applet 标记的 HTML 文件所在的一个目录。

方法 2：getImage(URL base, String target);

它从 base 所指定的目录中获取一个文件名为 target 的图像。其返回值是类 Image 的一个实例。

例如：Image tu = getImage(getDocumentBase( ), "石榴 .jpg");

表示从当前带有“<applet>”标记的 HTML 文件所在的路径获取图像文件“石榴 .jpg”。

此外，方法 getCodeBase( ) 返回描述 Applet 类文件本身所在的一个目录。此目录通常与 HTML 文件放在相同的目录中。

2. 播放音乐

要播放音乐，也要先获取该音乐文件的 URL 地址。

方法：getAudioClip(URL base, String target);

它从 base 所指定的目录中获取一个名为 target 的声音文件。其返回值是类 Audio Clip 的一个实例。例如：

```
AudioClip audio = getAudioClip(getDocumentBase( ),"fallingstar.mid");
```

表示从当前带有“<applet>”标记的 HTML 文件所在的路径获取声音文件“fallingstar.mid”。

播放声音片断（Audio Clips）最简单的方式是调用 Applet 的 play( ) 方法：

```
play(URL soundDirectory, String soundFile);
```

声音文件可以循环播放，调用 loop( ) 方法即可实现，而调用 stop( ) 方法则可暂停播放。

3. 线程

在计算机中完成一项任务，有时需要同时执行多个程序。例如，在显示一幅图像的同时播放背景音乐，这就需要同时执行显示图像和播放音乐两个程序。

通常我们将正在执行的一个程序称作一个进程。一个进程可以由一个或多个线程（Thread）构成。线程是程序进程里单一而连续的控制流程，就像线一样有头有尾。Java 的

线程就是程序进程中的一段程序代码。同一个进程中的不同线程，可以共享一定的系统资源而不会增加系统的额外负担，因此线程是今后程序设计的发展取向。

例如，一段程序代码可以由多个线程共享，当两个以上的线程执行同一个类的实例代码时，它们不仅可以共享相同的代码，还可以共享相同的数据，因为面向对象的程序代码和数据是相互独立的。

Java 支持多线程编程，利用线程进行编程的一般步骤如下：

（1）创建线程

在 Java 中创建线程一般有两种方法：

其一，通过创建一个 Thread 类封装在 java.lang 包中的对象直接产生线程，然后通过该对象来实现对线程的控制。这种方法适用于 Java 的 Application 应用程序的设计。例如：

```
Thread td = new Thread( );
```

其二，通过继承 Applet 类，安装 Runnable 接口，间接创建线程。这种方法适用于 Java 的 Applet 小程序的设计。因为 Java 的类只能继承一个类，而 Applet 程序本身已经继承了 Applet 类，就不能再继承其他的类（如 Thread 类）了，因此必须采用间接的方式（如接口）创建线程。

（2）启动线程

一个新创建的线程并不会自动运行，要使它运行，首先要调用它的 start( ) 方法，使线程处于可运行状态，然后再调用 run( ) 方法才行。

（3）暂停或终止线程

一个线程既可以通过调用 Thread.sleep( ) 方法暂停一段时间，也可以通过调用 Thread.stop( ) 方法终止线程。

Thread.sleep( ) 是一个静态方法，它的参数指定线程的最小休眠时间（以毫秒为单位），除非线程因为中断而提早恢复执行，否则它不会在这个时间段内恢复执行。

调用 interrupt( ) 方法可以使线程中断，并发出一个 InterruptedException 异常指令，可以使用 try 和 catch 结构抛出异常。

例如，下列代码中使用 try 和 catch 结构抛出异常。其中语句 Thread.sleep(200) 可以使线程暂停 200 毫秒。

```
try
{
Thread.sleep(200);
}
catch (InterruptedException e) { }
```

这段程序经编译成 class 文件后，嵌入到 HTML 文件中，在浏览器上看到的效果是，首先在网页上加载一幅图像，然后利用线程播放背景音乐（MIDI 文件）。图 2-7-6 所示是利用小程序浏览器 Appletviewer 浏览的一段程序的运行的结果。

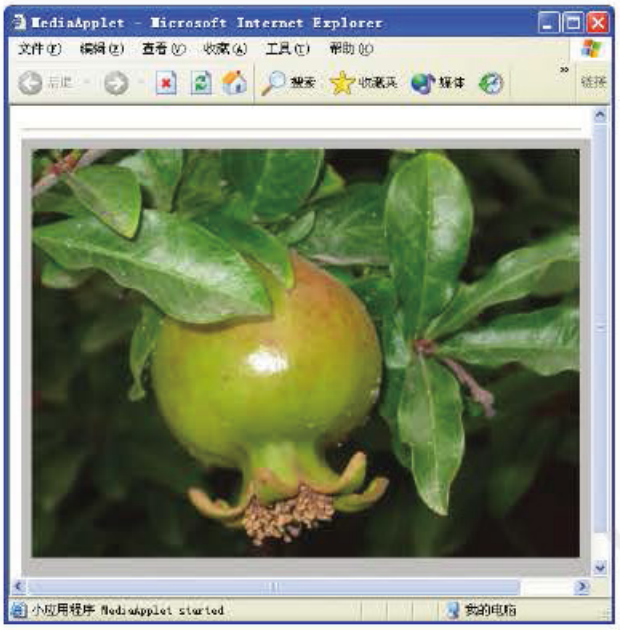


图2-7-6 加载图像并播放背景音乐

这段程序的完整代码如下：

```
// 显示图像、播放背景音乐
import java.awt.*;
import java.applet.*;
public class MediaApplet extends Applet implements Runnable
{
    Thread audioThread = null;      // 声明线程 Thread 对象
    String fileName;                // 声明 String 变量，表示音频文件名
    private AudioClip audio;        // 声明 AudioClip 类的 audio 对象
    Image tu;                       // 声明 Image 对象
    public void init( )             // 初始化 Applet
    {
        tu = getImage(getDocumentBase( ), "石榴 .jpg");
        // 从 HTML 文件中的参数获得音频文件名
        fileName = getParameter("fileName");
        if(audioThread == null)
        {
            audioThread = new Thread(this);      // 初始化线程
            audioThread.start( );                // 启动线程
        }
    }
    public void paint(Graphics g)
    {
```

```
        g.drawImage(tu, 10, 10, this);          // 显示图像
    }
    public void destory( )                      //Applet 结束
    {
        audio.stop( );
    }
    public void run( )          // 运行线程，加载并播放音频
    {
        audio = getAudioClip(getDocumentBase( ), fileName);
        audio.loop( );          // 重复播放
    }
}
```

要了解代码的功能，可以参看程序中的注释。将上述程序编译成 MediaApplet.class 文件，然后加入 HTML 文件中，程序代码如下：

```
<html>
   <head>
   <title>MediaApplet</title>
   </head>
   <body>
      <hr>
      <div align=center>
      <applet code=MediaApplet.class width=250 height=80>
      <param name="fileName" value="fallingstar.mid">
      </applet>
      </div>
   </body>
</html>
```

2　借用程序美化自己的网页。

你一定见过很多绚丽多彩的网页，是不是常被它的精彩特效所吸引？这些特效通常能用 Applet 来实现。因特网上有很多供免费下载的、能实现 Applet 特效的 Java 程序代码，只要对其稍加修改，就可以把它们插入到自己制作的网页中使用。

下面的程序代码演示出一段特效文字，如图 2-7-7 所示。读一读程序，对它进行适当的修改，并嵌入到自己创作的网页代码中，然后上传到电子学习档案袋“我的作品”中，并以“定稿”方式发布。比一比，看谁制作出来的特效最吸引人。

图2-7-7　特效文字

```
// 特效文字演示
import java.awt.*;
import java.applet.*;
// 定义基本类，实现与线程接口
public class FancyText2 extends Applet implements Runnable
{                                                    // 声明变量
    String s = null;
    int direct = 1;
    int Hrad = 12;
    int Vrad = 12;
    Thread thread = null;
    char words[ ];
    int phase = 0;
    Image offI;
    Graphics offG;
    Color colors[ ];
    private Font f;
    private FontMetrics fm;
    // 初始化
    public void init( )
    {
        String param = null;
        s = getParameter("word");                    // 读取 html 文件中的字符（参数）
        setBackground(Color.black);                  // 设置背景色
        words = new char[s.length( )];               // 将字符串内容读入数组
        s.getChars(0,s.length( ),words,0);           // 读取第一个字符
        // 使用双缓冲区的方法创建离屏图像区域和环境
        offI = createImage(getSize( ).width,getSize( ).height);
        offG = offI.getGraphics( );
        // 设置字体
        f = new Font(" 隶书 ",Font.BOLD,60);
        fm=getFontMetrics(f);
        offG.setFont(f);
        // 为每个字符设定颜色
        float h;
```

```
    colors = new Color[s.length( )];
    for (int i = 0; i < s.length( ); i++) {
          h = ((float)i) / ((float)s.length( ));
          colors[i] = new Color(Color.HSBtoRGB(h,1.0f,1.0f));
    }
}
// 启动线程
public void start( )
{
    if(thread == null)
    {
       thread = new Thread(this);
       thread.start( );
    }
}
// 终止线程
public void stop( )
{
    if (thread != null)
    {
       thread.stop( );
       thread = null;
    }
}
// 运行线程，隔一定时间调用一次 repaint( )
public void run( )
    {
       while (thread != null)
       {
          try {
                Thread.sleep(200);
             }catch (InterruptedException e){ }
          repaint( );
       }
    }
```

```
    public void update(Graphics g)
    {
       int x, y;
       double ang;
       // 用黑色填充绘图区域，实现清屏
       offG.setColor(Color.black);
       offG.fillRect(0,0,getSize( ).width,getSize( ).height);
       phase+=direct;
       phase%=8;
      // 绘制要显示的下一屏图像
       for(int i=0;i<s.length( );i++)
       {
          // 确定下一个字符位置
          ang = ((phase-i*direct)%8)/4.0*Math.PI;
          x = 20+fm.getMaxAdvance( )*i+(int) (Math.cos(ang)*Hrad);
          y = 60+(int)(Math.sin(ang)*Vrad);
          // 确定下一个字符颜色
          offG.setColor(colors[(phase+i)%s.length( )]);
          // 画出一个字符
          offG.drawChars(words,i,1,x,y);
          }
       paint(g); // 调用 paint( ) 方法
    }
    public void paint(Graphics g)
    {
       g.drawImage(offI,0,0,this);
    }
}
```

Java可视化集成开发环境简介

Java可视化集成开发环境（IDE）是利用图形用户界面将代码编辑、编译、调试、运行以及在线帮助汇集于一体的程序开发环境。在此开发环境中，可以方便地通过对菜单、工具栏操作以及拖放对象，生成所见即所得（可视化）的应用程序界面。

目前可视化集成开发环境呈现出百花齐放的局面，这些开发环境之间虽然有较大的差别，但是也都有一些共同的地方。例如，包含一个源代码编辑器，拥有图形开发界面，调试器支持各种JDK版本，

并支持多线程调试，提供在线帮助，等等。

下面介绍两种常见的 Java 可视化集成开发环境。

◆ JBuilder

它是 Borland 公司提供的一个 Java 可视化集成开发环境。JBuilder Personal 版本可免费下载。

◆ Sun One Studio

它是 Sun 公司提供的一个跨平台（Linux、Windows）的开发工具，能最先为新的 Java 技术（J2EE，J2SE）提供支持。这是其他的产品所无法比拟的。

Sun One Studio 的 Community 版也可以免费下载，但需要先注册成会员。

此外，常见的 Java 可视化集成开发环境还有 Visual Cafe for Java，Visual Age for Java 等等。

1．下面是一段绘制图形的代码段，请在此基础上编写完整的程序，并显示运行结果。

```
public void paint(Graphics g)
{
        g.setColor(Color.lightGray);              // 设置颜色（浅灰）
        g.fill3DRect(20,70,50,50,true);           // 画三维矩形
        g.fill3DRect(100,70,50,50,false);
        g.draw3DRect(20,150,50,50,true);
        g.draw3DRect(100,150,50,50,false);
        g.setColor(Color.blue);
        g.drawOval(140, 100, 200, 100);           // 画蓝色空心椭圆
        g.setColor(Color.red);
        g.fillOval(220, 40, 60, 60);              // 画红色实心椭圆
}
```

2．打开 JDK 安装目录中的演示文件“jdk\demo\jfc\java2D\Java2Demo.html”，可以看到如图 2-7-8 所示的页面。请依次单击各个标签，欣赏 Java 实现的精彩效果，体会 Java 强大的图形与动画设计功能。

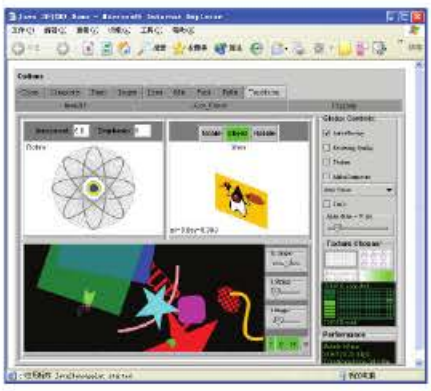


图2-7-8　Java2Demo.html的页面

3．上网搜索 Applet 特效，并下载应用到自己制作的网页上。

4．编写程序，演示小球做平抛运动的轨迹。

5．打开 JDK 安装目录中的演示文件夹“jdk\demo\applets”，可以看到 JDK 提供的许多 Applet 源程序代码，如图 2-7-9 所示。

例如，打开 Clock 目录，双击 example1.html 文件，就可以运行。试分别打开各个文件夹，浏览 html 文件，感受 Applet 的作用。

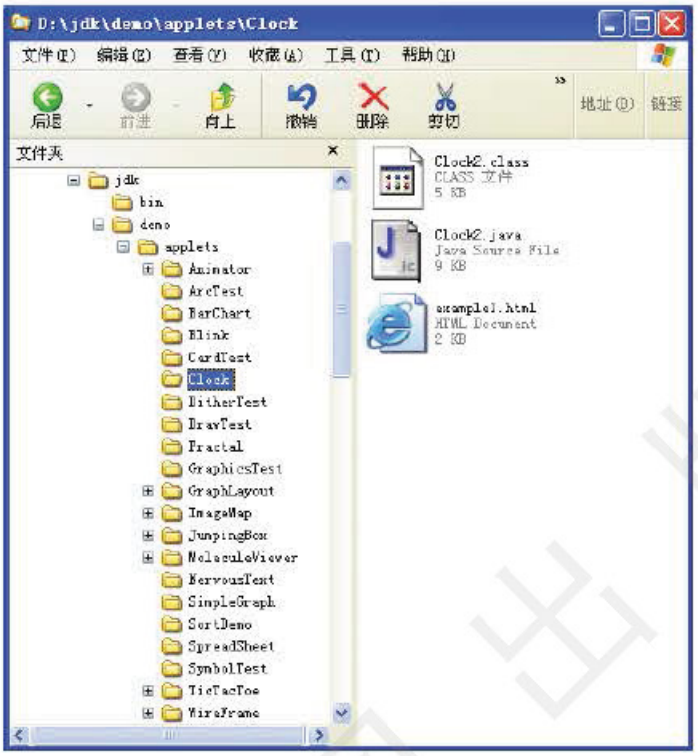


图2-7-9 JDK演示文件夹

第三单元·算法与问题的解决

“欲穷千里目，更上一层楼。”只有站在更高的位置，才可以看到更远的目标，算法的设计与应用就是我们展望远大编程目标的阶梯。

“算法 + 数据结构 = 程序设计”，同学们只要学会设计数据结构与算法，就可以编写出高效率的程序。

也许有人要问，计算机运行速度发展那么快，为什么还需要刻意设计高效率的程序？原因很简单，人类的雄心与能力是一起增长的，技术进步再快也快不过人们对需求的增长。计算速度和存储容量上的革新仅仅提供了处理更复杂问题的有效工具，但永远也满足不了人们的需求，所以高效率的程序永远不会过时。或许，我们开发的软件会表现出很多不如人意之处，但不要埋怨计算机或系统环境差，而应考虑设计一种更快、更好的算法，这才是解决问题的实质。

第一节　解析法与问题的解决

利用解析法设计算法，实质就是构造解题的数学模型。可以构造的数学模型通常有三类：(1) 直接利用“解析式”(或称公式) 解决问题；(2) 虽无法建立解析式，但能找出解决问题的一些关系式，用“递推”“迭代”等算法反复调用这个表达式最终解决问题；(3) 利用计算机速度快、精度高的优势，采用“模拟”的方法求解。本节将通过典型问题，分别介绍解析法的三种实现过程。

解决问题的关键是通过深入分析，将复杂问题简化、抽象，然后构建合理的数学模型，并选择适当的算法。

解析法是最基本的算法之一。利用解析法设计算法，实质就是构造数学模型。问题的类型不同，构造出的数学模型也不同。它可以是代数模型，也可以是几何模型，还可以是概率模型，等等。

利用解析法解决问题的一般思路如下：

一　代数模型问题

生活中许多问题都可以用解析式（或公式）来解决，直接用已知条件为变量赋值，即可求解。

1　制作矩形框。

①问题的提出

用一根长度为 l 的铁丝，制作一个面积为 s 的矩形框，要求计算出满足这种条件的矩形的高 h 和宽 w，如图 3-1-1 所示。

图3-1-1　矩形框

②问题分析

由已知条件，可以列出如下方程：

$$\begin{cases} 2w+2h=l & (\text{Ⅰ}) \\ s=wh & (\text{Ⅱ}) \end{cases}$$

由（Ⅰ）得

$$w=\frac{l-2h}{2} \quad (\text{Ⅲ})$$

将（Ⅲ）代入（Ⅱ）得：

$$s=\frac{l-2h}{2}h$$

化简后整理得：

$$2h^2-lh+2s=0$$

设 a=2，b=−l，c=2s，得 $ah^2+bh+c=0$（$a \neq 0$）。

这样就建立了解决问题的数学模型，亦即求解关于未知数 h 的一元二次方程的根。

③算法描述

用 d 表示判别式，即 $d=b^2-4ac$。若 d<0，则方程无实根；若 d=0，则方程有两个相等的根；若 d>0，则方程有两个不同的实根。

算法描述如图 3-1-2 所示。

<table>
<tr><td colspan="3">a←2,b← −l,c←2s</td></tr>
<tr><td colspan="3">d←b*b−4*a*c</td></tr>
<tr><td colspan="2">false　　d=0?</td><td>true</td></tr>
<tr><td rowspan="4">输出无解</td><td>false　d>0?　true</td><td rowspan="2">h← −b/(2*a)</td></tr>
<tr><td>h1←(−b+√d)/(2*a)
h2←(−b−√d)/(2*a)</td></tr>
<tr><td>w1←(l−2h1)/2</td><td rowspan="2">w←(l−2h)/2</td></tr>
<tr><td>w2←(l−2h2/2)</td></tr>
<tr><td colspan="2">输出w1,h1,w2,h2</td><td>输出w,h</td></tr>
</table>

图3-1-2　矩形框问题框图

④编程实现

试根据上述框图编写程序，可参考如下程序代码：

```
// 这是一个能计算一元二次方程的根的 Java 程序
import java.io.*;
public class E2root
{
    public static void main(String args[ ]) throws IOException
    {
```

```
double h,h1,h2,w;                    // 声明变量
// 使用缓冲区 (BufferedReader) 从文本数据流读取文本数据
InputStreamReader reader=new InputStreamReader(System.in);
BufferedReader input=new BufferedReader(reader);
System.out.print(" 请输入铁丝长度 :");
String s1=input.readLine( );                    // 从键盘输入铁丝长度
double l=Double.parseDouble(s1);          // 将字符串 s1 转换为实数
System.out.print(" 请输入矩形面积 :");
String s2=input.readLine( );                    // 从键盘输入矩形面积
double s=Double.parseDouble(s2);          // 将字符串 s2 转换为实数
double a=2,b=-l,c=2*s;          // 把数据转换为一元二次方程的系数
double d = b*b-4*a*c;           // 计算判别式 d=b²-4ac
if (d == 0)
{
    h=-b/(2*a);                  // 求出公共根 : 矩形高
    w=(l-2*h)/2;                 // 求出矩形宽
    System.out.println(" 矩形高 : "+h);       // 输出矩形高
    System.out.println(" 矩形宽 : "+w);       // 输出矩形宽
}
else
{
    if (d > 0)                   // 如果 d>0 ，那么求出两个实根
    {
        h1 = (-b + Math.sqrt(d)) / (2*a);
        h2 = (-b - Math.sqrt(d)) / (2*a);
        w1 = (l-2*h1)/2;
        w2 = (l-2*h2)/2;
        System.out.println(" 第 1 组解    矩形高 : "+h1+" 矩形宽 : "+w1);
        System.out.println(" 第 2 组解    矩形高 : "+h2+" 矩形宽 : "+w2);
    }
    else            // 如果 d<0，输出无解
        {
            System.out.println(" 问题无解 !" ); // 输出 " 问题无解 "
        }
}
}
```

2　中国人口问题。

①问题的提出

根据国家统计局2019年2月28日发布的“2018年国民经济和社会发展统计公报”，2018年末全国内地总人口139 538万人，比上年末增加530万人。全年出生人口1 523万人，出生率为10.94‰；死亡人口993万人，死亡率为7.13‰；自然增长率为3.81‰。目前有专家认为，从中长期考虑，中国应逐步调整人口政策。

以139 538万人为基础数据，假设每年的自然增长率为4.0‰，请你计算20年之后我国的总人口数。当每年的自然增长率为4.5‰或3.5‰时，20年之后我国总人口分别是多少？

②算法分析

计算n年之后的人口总数，公式是：

$$a=b\times(1+r)^n$$

其中，a为人口总数，b为基础人口数，r为年自然增长率。

计算总人口时，以“万人”为基本单位。可使用下列表达式计算，将a四舍五入为整数：

```
Math.floor(a+0.5)
```

③编程实现

程序如下：

```
import java.io.*;
public class Renkou
 {
  public static void main(String args[ ]) throws IOException
     {
       String s;
            // 使用缓冲区 (BufferedReader) 从文本数据流读取文本数据
        InputStreamReader reader=new InputStreamReader(System.in);
        BufferedReader input=new BufferedReader(reader);
        System.out.println(" 请输入自然增长率 :"); // 提示输入信息
        s=input.readLine( );              // 从键盘输入一个字符串
        double r=Double.parseDouble(s);
        double b=132802;     // 声明变量 b（总人口），并初始化
        int n=20;                // 声明变量 n（年限），并初始化
        double a=b*Math.pow((1+r),n);                  // 计算人口
        a=Math.floor(a+0.5);                           // 四舍五入
        System.out.println(" 共计 : "+a+ " 万人 " );   // 输出结果
     }
}
```

如果打算使程序能同时计算出多组数值，只需要添加循环语句即可，试试看吧！

二 递推模型问题

有些问题可以用递推关系式表示。这类问题无法将已知条件代入公式一次性完成求解，而需要多次使用这个公式，直到最终求出解，这种方法叫做递推或迭代法求解。递推法在解决问题的过程中是通过前面一些量依次推出后面的量的算法。如果每次操作都是在前一次的基础上进行，那么这种算法叫做迭代。累加器使用的算法就是迭代。递推往往可以表达为迭代的形式。递推和迭代都可以通过循环来实现。

任务3 求兔子繁殖的数量。

①问题的提出

意大利数学家菲波那契（Fibonacci）在1202年出版的《Liber Abaci》一书中提出了一个很有趣的数学问题：

一个人把一对小兔子放在四面都有围墙的地方，假定一对小兔子经过一个月以后就能够长大成为一对大兔子，而一对大兔子经过一个月后又能够生出一对小兔子，那么一年后总共有多少对兔子？

②问题分析

用○表示一对小兔子，用◎表示一对大兔子，用f_n表示n个月后的兔子数，兔子繁殖情况如图3-1-3所示。

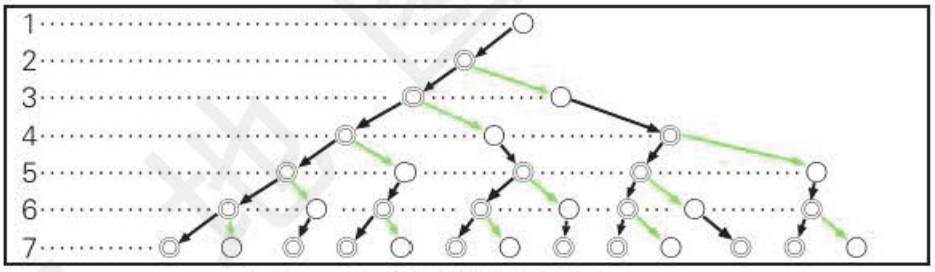


图3-1-3 兔子繁殖示意图

n与f_n的关系可以表示为：

n	1	2	3	4	5	6	7	8	9	10	11	12
f_n	1	1	2	3	5	8	13	21	34	55	89	144

观察数列：1，1，2，3，5，8，13，21，34，55，89，144，…可以归纳出下列规律：

$$\begin{cases} f_1=1 \\ f_2=1 \\ \cdots\cdots \\ f_n=f_{n-1}+f_{n-2}\ (n \geqslant 3) \end{cases}$$

数学家们把符合上述规律的数列叫做菲波那契数列，以纪念最先发现这个数列的数学家。

具体到本题，数列即为：

1，1，2，3，5，8，13，21，34，55，89，144，…

这个数列在数学、物理、化学、生物学中是经常出现的。

菲波那契数列就是递推数列，$f_n = f_{n-1}+f_{n-2}$ 是该数列的递推公式，$f_1=1$，$f_2=1$ 称为初始条件。要求出 f_n 的值，必须先求出 f_{n-1} 与 f_{n-2} 的值。

如果给出几个初始值，就可以求出任意的 f_n 值。

③算法描述

由所给递推关系知 $f_1=1$，$f_2=1$。根据递推关系，第一步，由 f_1 和 f_2 可求出 f_3；第二步，由 f_2 和 f_3 可求出 f_4……；第 (n−2) 步，由 f_{n-2} 和 f_{n-1} 可求出 f_n。

上述过程可使用循环来实现：

```
输入：正整数 n
输出：Fibonacci 数列的头 n 个元素
指令：
        输入 n；
        f1 ← 1；
        f2 ← 1；
        i ← 3；
        while(i<=n)
        {
            f3 ← f1+f2；
            输出 (f3)；
            f1 ← f2；
            f2 ← f3；
            i ← i+1；
        }
```

④编程实现

```
// 兔子繁殖数量问题
public class Fibonacci0
{
    public static void main(String args[ ])
    {
        int n = 12;
        int f1 = 1; int f2 = 1;          // 第一、二个月各一对
        int f3；
        System.out.println(" 第 "+1+" 个月：  "+f1+"  对兔子 ");
        System.out.println(" 第 "+2+" 个月：  "+f2+"  对兔子 ");
        int i=3;
        while(i<=n)
        {
 f3=f1+f2;     // 从第三个月开始，每月兔子数为前两个月之和
```

```
            System.out.println(" 第 "+i+" 个月 : "+f3+" 对兔子 ");
            f1=f2;          // 把 f2 的值赋给 f1
            f2=f3;          // 把 f3 的值赋给 f2
            i=i+1
        }
    }
}
```

程序经编译后运行，结果如图 3-1-4 所示。

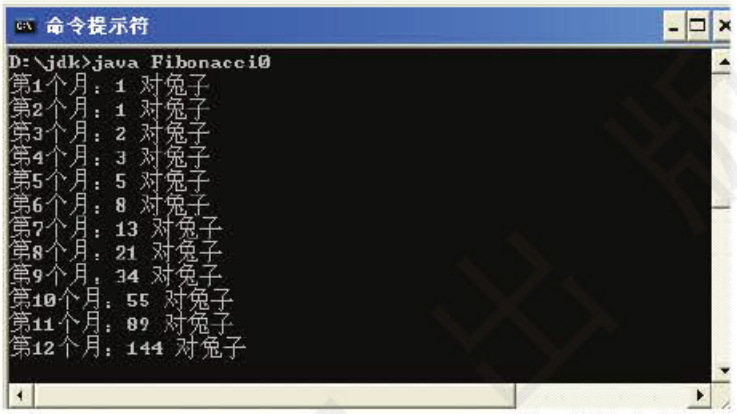


图3-1-4 兔子繁殖问题的运行结果

⑤使用数组求解

上述程序中，通过语句“f3=f1+f2”实现了迭代算法。本例我们还可以利用数组实现递推算法。程序如下：

```
// 兔子繁殖数量问题
public class Fibonacci1
{
    public static void main(String args[ ])
    {
        int n = 12;
        int a[ ]=new int[12];    // 定义数组 a，存放每月的兔子对数
        a[0] = 1; a[1] = 1;      // 第一、二个月各一对
        System.out.println(" 第 "+1+" 个月 : "+a[0]+" 对兔子 ");
        System.out.println(" 第 "+2+" 个月 : "+a[1]+" 对兔子 ");
        for (int i=2;i<n;i++) // 从第三个月开始，每月兔子数为前两个月之和
        {
            a[i] = a[i – 1] + a[i – 2];                    // 递推
            System.out.println(" 第 "+(i+1)+" 个月 : "+a[i]+" 对兔子 ");
        }
    }
}
```

三　概率模型问题

许多实际问题无法用一个简单的公式来描述。例如，抛掷一枚硬币，落地后，朝上的一面可能是正面，也可能是反面，要事先做出准确的判断是不可能的。这类现象有一个共同的特点：在基本条件不变的情况下，一系列的试验在某一可能范围内的不同结果，呈现一种偶然性，这种现象称为随机现象。在一定条件下，可能发生也可能不发生的事件称为随机事件。这种随机事件的数学模型叫做概率模型。

随机模拟，就是用计算机产生的随机数序列来模拟随机事件中的随机序列，也就是模拟真实事物的变化。它能解决在实践中难以解决或不易解决的随机事件问题。

Java 语言定义了一种随机方法：Math.random( ); 用来产生 [0,1) 之间的随机小数。

任务 4　小学生加法运算题。

①问题的提出

小明的表弟刚上小学，这几天临时住在小明家，这样小明就多了一个任务：每天给表弟出 50 道两位正整数加法的数学题。第一天他兴致勃勃地完成了任务，同时还批改了表弟的作业，但第二天他就感觉枯燥了。看着计算机，小明忽然灵机一动：何不设计一个程序，让计算机帮自己完成任务？

②问题分析

利用随机函数让计算机产生两个两位正整数，并显示在屏幕上，等待用户输入答案，然后判断数据的正确性，正确可得 2 分，否则不计分。利用循环产生 50 次，最后计算总分并显示出来。考虑到小学生的特点，在显示总分时可根据分数适当增加些“鼓励”的话语。

两个两位正整数都是随机产生的。利用 Math.random( ) 方法产生 [m,n) 范围内的随机整数的表达式如下：

```
(int)(Math.random( )*(n-m)+m)
```

③编程实现

程序如下：

```
import java.io.*;
public class Sizeyunsuan
{
public static void main(String args[ ]) throws IOException
{
int n=50;                    // 设置题目数
int a,b,answ,key,score=0;
 String s;
 // 使用缓冲区 (BufferedReader) 从文本数据流读取文本数据
 InputStreamReader reader=new InputStreamReader(System.in);
```

```
BufferedReader input=new BufferedReader(reader);
System.out.println(" 本次练习有 50 道题，请仔细认真答题 ");
for(int i=1;i<=n;i++)
        { a=(int)(Math.random( )*90+10);   // 思考 a 和 b 的值
          b=(int)(Math.random( )*90+10);
          answ=a+b;
          System.out.print(" 题 "+i+"   "+a+" + "+b+" = ");
          s=input.readLine( );                    // 从键盘输入一个字符串
           key=Integer.parseInt(s);
          if(key==answ)
                score=score+2;
         }
System.out.println(" 你的分数为 ："+score);
if(score>=90)
        System.out.println(" 你很棒，继续努力！ ");
        else
            if(score<60)
           System.out.println(" 成绩不理想，加油！ ");
}
}
```

以十道题为例，某一次随机产生的运行结果如图 3-1-5 所示。

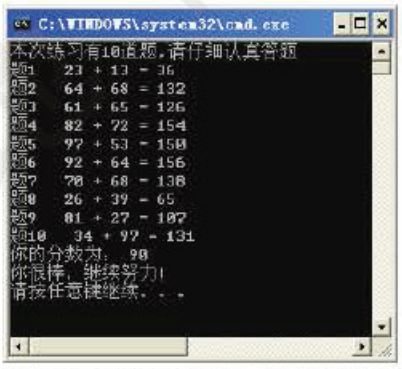


图3-1-5 加法运算某一次的运行结果

请思考以下问题，并完善程序：

1．程序中的 n 值是通过赋值完成的，能否通过程序运行时输入完成？

2．本程序只是加法运算，能否将其改为既有加法也有减法？要求加减法的出现是随机的，并且最后的结果不小于 0。

3．能否通过修改程序使其随机出现乘法运算，并且结果只能在 100 以内？

4．能否进一步完善程序，使其随机出现加、减、乘、除四则运算，并且除法的结果只能是整数？

试试看。如果你能完成以上 4 步，那么恭喜你，因为你已经真正开发出一个实用程序了。

小资料

π值是如何计算的

人们经过长期摸索和研究渐渐发现，圆不论大小，它的周长与直径之比是个不变的常数，这就是圆周率。这个重要发现吸引着许多数学家孜孜不倦地去计算圆周率的精确值。

在古代中国、埃及和希腊，都有不少学者计算过圆周率。公元1737年，数学家欧拉把π作为圆周率的符号，随后被普遍采用。

据史料记载，早在公元前3世纪，古希腊学者阿基米德已经想到用“逼近”的方法来计算圆周率。阿基米德利用“逼近”法，得到$3\frac{10}{71}<\pi<3\frac{1}{7}$的成就。

我国古代数学家在计算圆周率方面有着杰出的成就。大约在公元前2世纪西汉初年，我国的古算书《周髀算经》中就有“径一周三”的记载，即直径为1的圆，其周长约为3，也就是说圆周率为3，于是人们把“3”称为古率。

在魏晋时期刘徽割圆术的基础上，南北朝时期杰出的数学家和天文学家祖冲之（图3-1-6）使用更精密的方法，计算出圆周率π在如下两数之间：

$$3.1415926<\pi<3.1415927$$

并且还得到

$$\text{密率：}\frac{355}{113}\text{；}\quad\text{疏率：}\frac{22}{7}$$

图3-1-6　祖冲之

上式如果用小数表示，就是

密率：$\pi=3.1415929\cdots$；疏率：$\pi=3.1428\cdots$

祖冲之给出的圆周率准确到小数点后第7位，这个纪录在世界上保持了1000多年，直到16世纪，德国人奥托和荷兰人安托尼兹才重新发现密率，所以国内外多数人称$\frac{355}{113}$为“祖率”，以纪念祖冲之的伟大贡献。

1873年，英国数学家香克斯经过15年的艰辛计算把π计算到707位小数。

电子计算机的问世，使π的计算产生了新的突破。

1949年，当人们应用计算机计算π时，很快得到小数点后2037位。有人对香克斯的计算进行验证，发现他只算对了527位小数，后面的数字全错了。

1973年，法国数学家盖劳德花了23小时8分钟，把π计算到100万位小数。

1984年，日本国立东京大学两位教师用高速电子计算机把π计算到1001.3395万位小数，这在当时可算是奇迹。

1999年9月，日本东京大学教授应用并行超级计算机，用了37小时21分钟，把π计算到2061.5843亿位小数，创造了世界最高记录。如果把计算出的这个π的近似值打印在A4纸上，每页印2万位数，那么这些纸叠起来可高达五六百米。

古今中外数学家们计算π的近似值主要用三种方法：

其一，利用“正多边形逼近”的方法求出π的近似值。

“割圆术”的创造者刘徽发现圆的内接正6边形每条边长都等于半径，6条边长等于3倍的直径长。因此，他指出古率“3”事实上不是圆的“周率”，而是圆的内接正6边形的“周率”。于是，他另辟蹊径，把圆的内接正6边形每条边所对的弧长进行平分，然后把分出的6个点与圆的内接正6边形的6个顶点依次连接，得到一个圆的内接正12边形，这样更逼近圆的周长。刘徽用这种方法，相继5次增加圆的内接正多边形，作出了圆的内接正192边形，算得 π 的近似值为3.14。

其二，利用迭代法计算 π 的近似值。

利用几何方法计算 π 值太繁琐。为寻求更快的计算公式，世界各国的数学家经历了漫长而艰苦的探索。但直至400多年前，欧洲文艺复兴，科学技术飞跃，特别是新的数学工具出现之后，π 值的算法研究才获得突破，并使其有效位数得到明显的延伸。

第一个放弃几何方法而使用新方法获得结果的似乎是法国的韦特，他给出不断开方的公式：

$$\frac{2}{\pi}=\sqrt{\frac{1}{2}}\times\sqrt{\frac{1}{2}+\frac{1}{2}\sqrt{\frac{1}{2}}}\times\sqrt{\frac{1}{2}+\frac{1}{2}\sqrt{\frac{1}{2}+\frac{1}{2}\sqrt{\frac{1}{2}}}}\cdots$$

近300年来计算 π 近似值的主流方法是级数方法。它主要是根据 π 与三角函数的关系，把三角函数展开成级数来计算 π。

例如：德国的数学家莱布尼兹在1674年得到 π 的表达式：

$$\frac{\pi}{4}=1-\frac{1}{3}+\frac{1}{5}-\frac{1}{7}+\frac{1}{9}-\cdots\frac{1}{4n-3}-\frac{1}{4n-1}$$

科学家牛顿在1676年得到计算 π 值的表达式：

$$\frac{\pi}{6}=\frac{1}{2}+\frac{1}{2\times3}\times\frac{1}{2^3}+\frac{1\times3}{2\times4\times5}\times\frac{1}{2^5}+\frac{1\times3\times5}{2\times4\times6\times7}\times\frac{1}{2^7}+\cdots$$

还有其他一些表达式，如：

$$\frac{\pi}{6}=\operatorname{arctg}\frac{1}{\sqrt{3}},\quad \operatorname{arctg}x=x-\frac{x^3}{3}+\frac{x^5}{5}-\frac{x^7}{7}+\cdots$$

其三，利用蒙特卡洛法（Monte-Carlo Method）计算 π 的近似值。

法国数学家蒲丰1777年提出了随机投针的方法，利用此法求得 π 的近似值为3.142。1901年，意大利学者利拉查里尼投针3480次，得到 π 的近似值为3.1415929。

这个有趣的方法创立了数值计算中的随机近似算法——蒙特卡洛法。它是一种模拟随机事件发生的算法，根据模拟实验结果进行统计而得出结论。

利用蒙特卡洛法计算 π 的值，主要是通过计算一类曲边封闭图形面积求出的。

在一个边长为1的正方形中，以边长为半径，以一个顶点为圆心，在正方形上作1/4圆。随机向正方形内扔点，若落入1/4圆内则计数，用m表示。重复向正方形内扔足够多的点，将落入1/4圆内的计数除以总的点数n，其值就是 π 值1/4的近似值。

圆面积公式为 $S=\pi r^2$。当 $r=1$ 时，$S=\pi$，所以只要求出圆 $x^2+y^2=1$ 所包围的面积，就能求出 π 的值。为简便起见，可先求出如图3-1-7中涂色部分（1/4圆）的面积，然后乘以4即可。

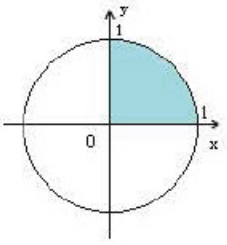


图3-1-7 单位圆

在以(0,0)、(1,0)、(1,1)、(0,1)为顶点的矩形内产生n个随机点。这些随机点的横坐标范围在0到1之间，纵坐标范围也在0到1之间，如果随机点的纵坐标值小于或等于 $\sqrt{1-x^2}$ 的值，就认为该点落在涂色范围内，否则认为该点落在涂

色范围之外。当n充分大时，可以认为：涂色面积 / 矩形面积 =m/n。

所以：

涂色面积 = 矩形面积 ×（m / n）。

1.个人所得税是国家对个人所得额征收的一种税。个人工资薪金所得采用超额累进税率，“高收入者多纳税，低收入者少纳税”是个人所得税的显著特点。我国1981年开征个人所得税，且只对外国公民征收；1986年开征城乡个体工商户所得税；1987年对我国公民开征个人收入调节税；1994年上述三税合并为现行个人所得税。2019年1月1日起个人工资薪金所得税率如下表：

级　数	全月应纳税所得额（减去5 000元后的余额）	税　率
1	不超过3 000元的部分	3
2	超过3 000元至12 000元的部分	10
3	超过12 000元至25 000元的部分	20
4	超过25 000元至35 000元的部分	25
5	超过35 000元至55 000元的部分	30
6	超过55 000元至80 000元的部分	35
7	超过80 000元的部分	45

上表中“全月应纳税所得额”是从月薪金收入中减去5 000元后的余额。例如，某人扣除各项基本费用后的月薪金收入8 520元，减去5 000元，应纳税所得额为3 520元，由税率表可知，其中3 000元税率为3%，另外520元税率为10%，所以此人应缴纳个人所得税：

3 000×3%+520×10%=142（元）

试根据上述资料编程计算个人所得税，并根据你的家庭成员的月收入求出每月需要缴纳的个人所得税。（注：个人所得税起征点或税率若有调整，可根据最新数据编程计算）

2．根据下列展开式（无穷级数）编写程序，计算正弦函数sinx的近似值。

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots\cdots$$

第二节　穷举法与问题的解决

穷举法也是问题求解常用的算法之一。本节通过“韩信点兵”问题以及逻辑推理问题的求解，介绍利用穷举算法解决问题的基本思路和设计方法。

穷举又称为枚举。穷举法的基本思想是：依据题目的已知条件，确定答案的大致范围，在此范围内对所有可能的情况逐一验证，直到全部情况得到验证。若某种情况符合题目的全部条件，则为本题的答案；若全部情况都不符合题目的条件，则本题无答案。

适合用穷举法求解的问题是：问题的可能答案数量有限而不是无限，并且这些可能的答案必须是事先知道的。在程序设计中，穷举法主要通过循环语句来实现。

穷举法的特点是算法简单，但是当需要穷举的数据量很大时，运行程序所花费的时间也就非常大，此时应该考虑优化算法。

一　“韩信点兵”

1. 问题提出

韩信是我国秦末汉初的著名军事家，历史上有关韩信用兵的传说很多。韩信统率过千军万马，据说他对手下士兵的数目了如指掌。他统计士兵数目有个独特的方法：先令士兵排成 5 列纵队，结果余 1 人；接着，又命令士兵排成 6 列纵队，结果余 5 人；随后，再命令士兵排成 7 列纵队，结果余 4 人；最后，命令士兵排成 11 列纵队，结果余 10 人。这样他便知道部队的实际人数了。后人把他这种统计士兵人数的方法称为“韩信点兵”，历史上还称之为“鬼谷算”或“隔墙算”。

2. 问题分析

此类问题可以理解为：一个数除以 5 余 1，除以 6 余 5，除以 7 余 4，除以 11 余 10，求符合这些条件的数中最小的数。

我们先用数学方法分析此题的解法。

第一步，求 5，6，7，11 的最小公倍数：$M=5\times6\times7\times11=2310$。

第二步，分别找出符合每个已知条件的最小数。

①除以 5 余 1 的数。要在 6，7，11 的公倍数中去找。6，7，11 的最小公倍数是 462，就用 462 来试。

$462\div5=92\cdots\cdots2$

因为 462 除以 5 的余数是 2，不符合已知条件，就继续用 462 的 2 倍数来试，直到找到余数是 1 的 462 的倍数，本题中当 462 扩大 3 倍时，除以 5 余 1，即：

$462\times3\div5=277\cdots\cdots1$

则 462 × 3=1386 符合“除以 5 余 1”的条件。

②除以 6 余 5 的数。要在 5，7，11 的公倍数中去找。5，7，11 的最小公倍数是 385。

385 ÷ 6=64……1

条件要求除以 6 余 5，要满足余数是 5 的条件，就必须使被除数、商、余数同时扩大 5 倍。385 × 5=1925，1925 ÷ 6=320……5。

则 1925 符合“除以 6 余 5”的条件。

③除以 7 余 4 的数。要在 5，6，11 的公倍数中去找。5，6，11 的最小公倍数是 330。

330 ÷ 7=47……1

条件要求除以 7 余 4，要满足余数是 4 的条件，被除数、商、余数必须扩大 4 倍。330 × 4=1320。

1320 ÷ 7=188……4

则 1320 符合“除以 7 余 4”的条件。

④除以 11 余 10 的数。要在 5，6，7 的公倍数中去找。5，6，7 的最小公倍数是 210。

210 ÷ 11=19……1

条件要求除以 11 余 10，要满足余数是 10 的条件，就必须使被除数、商、余数同时扩大 10 倍。210 × 10=2100，2100 ÷ 11=190……10。

则 2100 符合“除以 11 余 10”的条件。

第三步，将第二步中①②③④中求得的符合条件的数相加。

1386+1925+1320+2100=6731

结果“6731”符合题目中全部条件。

第四步，求出符合题意的最小解。

6731 虽然符合题意的全部条件，但并不是最小解，还要减去 5，6，7，11 的最小公倍数 2310 的倍数，因此求出的最小解为：

6731−2 × 2310=2111

从用数学方法分析该问题的过程来看，解决此题的思路应该是先抽象出一个数学公式，再用解析法编程求解。但是要抽象出一个解析式来，难度还是比较大的。

现在换一个角度来寻求解决方案，题意是“一个数除以 5 余 1，除以 6 余 5，除以 7 余 4，除以 11 余 10，求符合这些条件的数中最小的数”，那么，我们从最小的数开始一个一个地去试，只要满足四个条件，该数就是最终结果。这种做法就是穷举法。用穷举法求解此题，解题思路是可以的，但是如果要求人来做，难度会比抽象出一个解析式还大。计算机的优势之一就是具有高速计算的能力，因此我们就利用计算机的这一优势，让计算机一个一个去试。

3. 算法描述

由于解决此问题主要是求出满足已知条件的正整数，所以利用穷举法求解的关键是确定穷举范围，由题目条件可知穷举范围的最小值为 21（排成 11 列纵队，结果余 10 人），因此可以从 21 开始逐个检查正整数，检验是否满足条件。如果满足已知条件，说明已经找到解，则终止循环。此算法的流程图如图 3-2-1 所示。

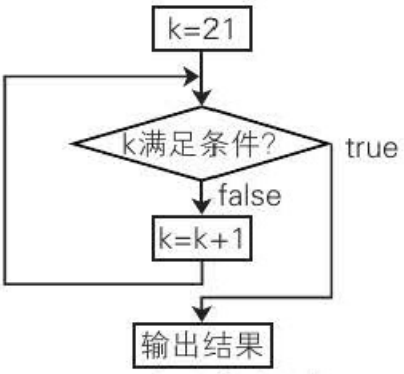


图3-2-1 穷举算法的流程图

由于不知道循环的次数，所以只能使用条件循环语句来实现。现用伪代码对算法描述如下：

```
输入：问题中的数据均为常数，故无输入数据
输出：满足条件的最小值
指令：
        k ← 21;
 /* 下面的逻辑表达式表示 k 同时满足：除以 5 余 1，除以 6 余 5，除以 7 余 4，除以 11 余 10*/
        while (!((k%5==1)&&(k%6==5)&&(k%7==4)&&(k%11==10)))
        {
         k=k+1;
        }
        输出结果 k;
```

4. 程序实现

从上述算法可以看出，这是一种当型循环，因此，在程序的循环体内一定要有改变条件表达式值的语句，如 k=k+1。具体的程序代码如下：

```
// 韩信点兵
public class Meiju1
{
   public static void main(String args[ ])
     {
        int k=21;                    // 从 21 开始，对 k 穷举
        while (!((k%5==1)&&(k%6==5)&&(k%7==4)&&(k%11==10)))
          {
             k++;                    // 当没有满足条件时，k 自动加 1
          }
         System.out.println("k="+k);      // 输出结果
     }
 }
```

图3-2-2 韩信点兵程序的运行结果

程序经编译后运行，结果如图 3-2-2 所示。

小资料　“韩信点兵”与“中国剩余定理”

18世纪中期欧洲数学家欧拉(L.Euler，1707-1783)、拉格朗日(J.Lagrange，1793-1813)等都曾对一次同余式问题进行过研究。德国著名数学家高斯(C.F.Gauss，1777-1855)在1801年出版的《算术探究》一书中提出了不定方程求解方法。我国的《孙子算经》叙述的也是一次同余式问题的解法。

“韩信点兵”的故事还是源于《孙子算经》。1852年英国人伟烈亚力将《孙子算经》传入欧洲，人们才知道“物不知数”的解法与数学家高斯的定理是一样的。中国人的研究领先欧洲1000多年。所以，人们将一次同余式问题的求解定理称为“中国剩余定理”或“孙子余数定理”。这个定理不但在世界古代代数领域是一个重要的定理，而且在近代数学和程序设计中也得到很广泛的应用。

“中国剩余定理”用现代数学语言来描述就是：

设 d_1，d_2，d_3，…，d_n 互为质数，如果数 x 分别被 d_1，d_2，d_3，…，d_n 除得的余数分别是 r_1，r_2，r_3，…，r_n，那么 x 可表示为

$$x=k_1r_1+k_2r_2+k_3r_3+\cdots+k_nr_n+kd$$

其中：k_1 是 d_1，d_2，d_3，…，d_n 的公倍数，而且被 d_1 除得的余数是1；

k_2 是 d_1，d_2，d_3，…，d_n 的公倍数，而且被 d_2 除得的余数是1；

……

k_n 是 d_1，d_2，d_3，…，d_n 的公倍数，而且被 d_n 除得的余数是1；

d 是 d_1，d_2，d_3，…，d_n 的公倍数，k 可以是任意整数，应根据具体问题确定。

二　逻辑推理问题

1. 问题提出

四位同学中有一位同学做了好事没有留名，老师问：“是哪一位同学做了好事不留名？”A说：“不是我。”B说：“是C。”C说：“是D。”D反驳说：“不是我！”已知四个人当中只有一个人说假话，试判断是哪一位同学做了好事没有留名。

2. 问题分析

这类题目通常叫逻辑判断题或逻辑推理题。求解这类题目时，首先要对复杂的逻辑关系进行分析、抽象、化简，用逻辑表达式来表示需要判断的条件，建立起数学模型；然后用穷举法把各种可能的情况列举出来；最后根据已知条件排除错误结果或选择正确结果。

3. 算法设计

用1，2，3，4分别表示A，B，C，D四个人，i代表做好事的人，则他们说的话可表示为：

A说：$i \neq 1$

B说：$i = 3$

C说：$i = 4$

D说：$i \neq 4$

由于做好事的人（i）肯定是A，B，C，D中的一个，那么就让i从1至4，一个一个地测试。用k统计说真话的人数，由于只有一人说假话，所以当k=3时，i的值即为做好事的人。

程序如下：

```
// 逻辑推理问题
class logic1
{
    public static void main(String args[ ])
    {
       for (int i=1;i<=4;i++)          // 在 1 ~ 4 中逐一穷举做好事的人
       {
            int k = 0;                 // 用 k 累计说真话的人数
            if (i!=1) k = k + 1;       // 如果 A 说真话，k 值加 1
            if (i==3) k = k + 1;       // 如果 B 说真话，k 值加 1
            if (i==4) k = k + 1;       // 如果 C 说真话，k 值加 1
            if (i!=4) k = k + 1;       // 如果 D 说真话，k 值加 1
            if (k==3)                  // 如果有三个人说真话，就输出结果
            System.out.println(i);
       }
    }
}
```

程序运行结果为 3，表明是 C 同学做好事。

寻找100以内的所有素数。

①使用穷举法输出 100 以内的所有素数

素数也叫质数，是指在大于 1 的整数中，只能被 1 和这个数本身整除的数，如：2，3，5，7，11。

调用下面的 isPrime( ) 方法可以判断一个数是否为素数。参数为正整数 n，返回值为 true 时，表示 n 是素数；返回值为 false 时，表示 n 不是素数。

```
// 建立 isPrime( ) 方法，判断一个数是否为素数
static boolean isPrime(int n)
{
    if (n<2) return false;          // 如果 n<2，那么 n 不是素数
    if (n==2) return true;          // 如果 n=2，那么 n 是素数
    // 穷举 2 ~ n-1 之间的所有数，如果能整除 n，则 n 不是素数
    for (int d=2;d<n;d++)
        if (n%d==0) return false;
    return true;                    // 否则 n 是素数
}
```

试编写完整的程序代码，调用 isPrime( ) 方法，输出 100 以内的所有素数。

②优化与完善算法

求素数的方法有很多种。上面利用素数定义编写的程序运行效率非常低，其中有不少地方值得改进。例如：

◆对于一个自然数 n，只要能被一个非 1 又非自身的数整除，那么，它肯定不是素数，所以不必再用其他数去除。

◆对于一个自然数 n，只要用小于 n 的素数去除就可以了。例如，如果它不能被 3 或 5 整除，那么它肯定不能被 3 和 5 的最小公倍数 15 整除，所以不必用 15 去除 n。

◆对于一个自然数 n，不必用 2 到 n-1 之间的所有素数去除，只要用小于等于 $\sqrt{n}$ 的素数去除就可以了。这一点数学上已有证明。

试根据上述方法对算法进行优化，重新编写程序求出 100 以内的所有素数。

素数与现代密码技术

许多数学难题常常与素数有关。素数的重要性早在古希腊时期就已为人们所认识，著名数学家欧几里德在其巨著《几何原本》第四卷中证明了算术基本定理：每一个大于 1 的自然数，或者是素数，或者可表示为若干素数的乘积。这种表示，若不计素数的排列次序的话，则是唯一的。

算术基本定理告诉我们，素数是构造自然数的基本元素，所有的自然数都是用素数构造起来的。将一个自然数进行素因子分解，几乎能得到有关这个数的全部信息。

在整个自然数数列中，越往后面，数列中每个元素是素数的几率就越小，这是稍有算术常识的人都会意识到的问题。那么在自然数家族中素数到底有多少个？有没有最大的素数呢？这是素数研究的基本问题之一。

欧几里德早已证明，在自然数家族中素数的个数不可能是有限的，即没有最大的素数。接下来的问题是如何寻找素数，怎样表示素数，怎样判定某个自然数是素数。素数一直是一个不解之谜。寻找素数的问题，几百年来不知吸引了多少数学家，他们呕心沥血，刻苦研究，至今仍未彻底解决。

1640 年，法国大数学家费马（P. de Fermat，公元 1601-1665）认为他找到了能计算出素数的公式，这就是有名的费马数：$F(n)=2^{2^n}+1$。

因为 $F(0)=2^{2^0}+1=3$，$F(1)=2^{2^1}+1=5$，$F(2)=2^{2^2}+1=17$，$F(3)=2^{2^3}+1=257$，$F(4)=2^{2^4}+1=65537$，这些数都是素数，推而广之，所以费马认为所有 F（n）都是素数。

然而，大约一个世纪之后，大数学家欧拉（LEuler，公元 1700-1783）推翻了费马的结论，他在 1731 年计算出 $F(5)=4294967297$，该数可被 641 整除。即：

$$4294967297=6700417\times 641$$

1971 年，美国人布里哈特和莫里森再次证明了费马结论的错误，他们利用加利福尼亚大学洛杉矶分校的一台计算机找到了 F（7）的两个素因子：

$F(7)=2^{2^7}+1=2^{128}+1$

$=340282366920938463463374607431768211457$

$=59649589127497217\times 5704689200685129054721$

由此可见，猜想常常是靠不住的，任何猜想都需要经过严格的论证。从上面的计算可以看出，要判断一个费马数是否是素数相当困难，要计算出费马数也不容易。

此外，整数的因子分解跟素数有关。

一般来说，要把一个整数分解因子，只能拿 2，3，4，……去试除。那么要试到什么时候为止呢？通常要试除到原数的一半为止。当然，实际上只要用其中的素数去试除就行了，且只要用不超过该数的平方根的素数去试除，就能得出该数能否分解因子的结论。

随着整数数字位数的增多，这种试除工作量就快速增大，即使是现代快速计算机也难以胜任。另一方面，大数的因子分解在某种意义上与寻找大素数联系在一起，这是一个难度很大的工作。

大素数与现代密码技术有着密切的关系。所有的现代密码体系都要使用计算机，一般都假定他人使用强大的计算机来分析有关密码的信息，所以系统应建立足够复杂的密码体系防止被攻破。

为了使所设计的密码体系足够安全，通常把它分成两部分：一个加密算法和一把“钥匙”（密码学上称为密钥）。加密算法是由一个计算机程序或一台专门设计的计算机来实现的。系统给信息加密，就是使用选定的“钥匙”（通常它是一个秘密选定的数）对信息进行编码，使得只有掌握这把钥匙的人才可能解码，解码后即可得到正确的原始信息。

需要保密的、用以解密的“钥匙”，实质上是经过特殊挑选的两个大素数，公开的、用于加密的钥匙就是这两个素数的乘积。解密需要对这个大合数进行因子分解。在信息有效期内，一般无法完成这个合数的因子分解，从而可保证信息的安全。但是随着算法的进步，分解几百位乃至上千位长度的大数的因子也许不再成为难题，那么，要保障数据通信的安全，就必须有更巧妙的方法。

实践与思考

1．一辆卡车违反交通规则，撞人后逃跑。现场有三人目击了该事件，但都没有记住车号，只记下车号的一些特征。甲说：车号的前两位数字是相同的；乙说：车号的后两位数字是相同的，但与前两位不同；丙说：车号的四位数刚好是一个整数的平方。请根据以上线索编写程序，求出车号。

2．我国现行流通的人民币共有 13 种面额，分别是：100 元、50 元、20 元、10 元、5 元、2 元、1 元、5 角、2 角、1 角、5 分、2 分、1 分。当你支付 100 元人民币用于购买低于 100 元（不含角、分）的商品时，要求找回尽可能少的纸币。请编写程序寻找一个解决方案。

3．在一个直角三角形中，三条边 a，b，c 的长度都为整数，且一条直角边 a 的长度已确定，斜边 c 的长度不能超过某数 n。请编写程序找出能满足条件的所有直角三角形。

第三节 递归法与问题的解决

“大事化小、小事化了”常用来化解生活中的矛盾，现在在程序设计中也得到体现。递归算法就是按照这种思路来处理问题的，找到一个递归关系式将大问题转化为小问题，将小问题的解决方案作为问题解决的出口，大问题也就“迎刃而解”了。本节通过阶乘的递归调用和梵塔问题的求解，学习使用递归法解决问题的基本方法。

自己调用自己的方法叫做递归。一个问题能够适用递归方法解决，必须要符合两个条件：

其一，一个规模较大的问题可以分解为若干个性质相同的规模较小的问题，这些规模较小的问题仍然可以进一步分解，分解出的新问题的解法和原问题的解法相同，只是处理对象的规模不同。

其二，必须有一个明确的结束递归的条件，不得无限递归。

以下两种情况适合用递归法解决：

◆函数定义是递归的。

◆解法是递归的。

从递归算法的结构来分析，设计递归算法就是要解决递归出口和递归体两个问题，即要确定何时到达递归出口，何时执行递归体，执行什么样的递归体。设计递归法算法的关键是保存每一层的局部变量并运用这些局部变量。

递归算法的设计可分为以下四个步骤：

①分析问题，分解出小问题；

②找出小问题与大问题之间的关系，判断递归条件，确定递归出口；

③定义递归方法（函数）；

④在程序中调用递归方法。

下面举例说明递归算法是如何实现的。

一 计算阶乘

1. 问题分析

阶乘的计算就是一种典型的递归问题。阶乘通常定义为：

$f(n)=n!=n\times(n-1)\times(n-2)\times\cdots\times3\times2\times1$

也可定义为：

$f(n)=n\times f(n-1)$

这是一种递归定义的方法。

递归必须有终止条件（又称边界条件），从数学知识中得知 0!=1，这就是阶乘的终止条件。所以，阶乘函数可定义为：

$$f(n)=\begin{cases} 1 & n=0 \\ n\times f(n-1) & n>0 \end{cases}$$

按照上述阶乘函数的定义，对算法描述如下：

```
设 f(n)=n!      //n 为自然数
如果 n=0,  则 f(n)=1
否则，  f(n)=n×f(n-1)
```

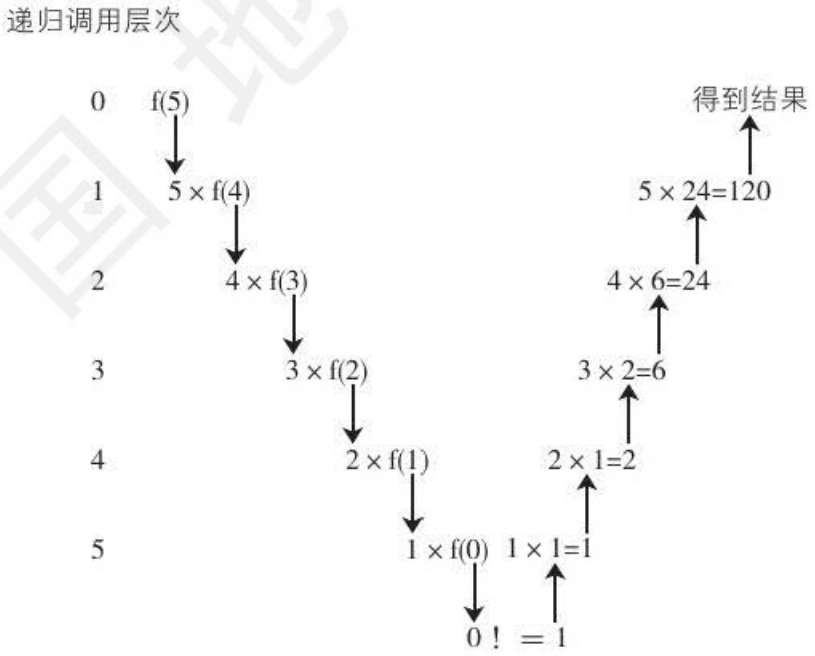
例如计算 5!，依次把 n = 5，4，3、2，1 代入 f(n)=n×f(n−1)，得：

$$\begin{aligned} 5! = f(5) &= 5\times f(4) \\ &= 5\times 4\times f(3) \\ &= 5\times 4\times 3\times f(2) \\ &= 5\times 4\times 3\times 2\times f(1) \\ &= 5\times 4\times 3\times 2\times 1\times f(0) \\ &= 5\times 4\times 3\times 2\times 1\times 1 \\ &= 120 \end{aligned}$$

2. 算法设计

为了计算 f(5) 的值，应计算 5×f(4)，所以要先计算 f(4) 的值；而要求出 f(4) 的值，得先求出 f(3) 的值；欲求 f(3)，得先求 f(2)；求 f(2)，得先求 f(1)；求 f(1)，得先求 f(0)。f(0) 的值为递归边界值，它是已知数 f(0)=0!=1。

上述递归调用过程表示如下：



从以上分析过程可以看出，递归可以分为两步：先不断调用，再依次退回。计算工作是在返回过程中进行的。

3. 编程实现

定义上述算法 f(n) 方法的代码如下：

```
static long f(long n)          // 定义方法 f(n), 利用递归计算 n!
      {
          if (n == 0)
           return 1;
        else
          return n*f(n-1);
        }
```

n 的值可以利用不同方法输入，调用上述 f(n) 方法的程序如下：

```
// 计算阶乘 n!
public class Factor2
{
    static long f(long n)          // 定义方法 f(n), 利用递归计算 n!
    {
        if (n == 0)
            return 1;
       else
            return n*f(n-1);
    }
    public static void main(String args[ ])
    {
        long n = Long.parseLong(args[0]); // 从命令行输入参数 n 的值
        for (long i=1; i<=n; i++)
        // 调用方法 f(n)，从 1 到 n 逐个输出 n! 的值
        System.out.println( "f( " + i + ")=" + f(i));
    }
}
```

程序经编译后运行，结果如图 3-3-1 所示。

图3-3-1　阶乘程序的运行结果

二 汉诺塔问题

1. 问题提出

汉诺塔问题是由一位法国数学家在19世纪提出的，直到现在，简化版的汉诺塔玩具在玩具店中仍然随处可见。汉诺塔的假设是：在一个铜板上，插着三根宝石针，分别被编号为a、b、c。在a针上，从下到上放置了由大到小的64个金片，要把这些金片借助b针，从a针全部移动到c针。移动规则是：一次只能移动一片，并且无论在哪一根针上，小片必须放置在大片上面。

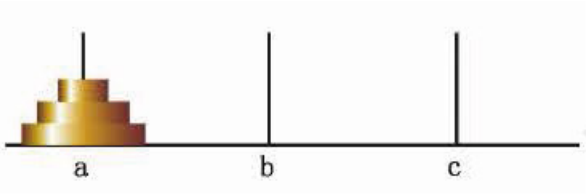


图3-3-2 汉诺塔的示意图

让我们开动脑筋、仔细分析，算一算移完全部的64片金片究竟需要多少时间。

2. 问题分析

金片在移动过程中有两个要求：

◆每次只能移动一个金片。

◆在一个针上任何时候都不能把大片放在小片上面。

我们把三根针分别称为a，b，c。开始时金片都放在a针上，经过起“中间站”作用的b针，最后移到c针上。

为了寻找移动规律，不失一般性而又简化问题，可以两个片金片为例，即设金片的个数n=2，小的编号为1，大的编号为2。

现在要把两片金片从a经b移至c，步骤如图3-3-3所示。

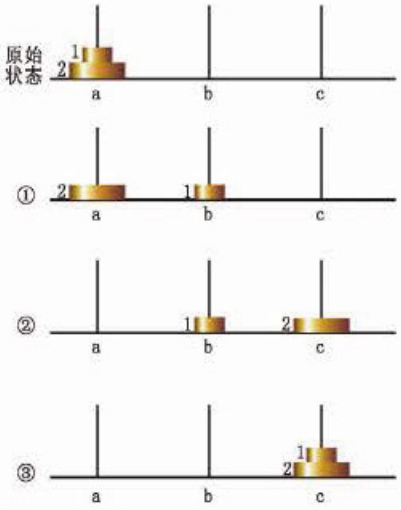


图3-3-3 金片移动步骤示意图

上述过程可描述为：move(2，a，b，c)。它由如下三步组成：

将1号金片从a移至b　　move 1：a→b

将2号金片从a移至c　　move 2：a→c

将1号金片从b移至c　　move 1：b→c

现在把问题一般化，把n片金片(n>2)从a经b移至c。可把这一过程描述为：

move(n, a, b, c)

我们可以把上面的n-1片金片“捆”在一起当作一片金片。现在移动步骤如下：

将(n-1)片金片从a经c移至b

将n号金片从a移至c

将(n-1)片金片从b经a移至c

上述过程可描述为：

move(n−1, a, c, b)

n: a → c

move(n−1, b, a, c)

现在已经把 n 片金片移动过程简化为把（n−1）片金片从 a 经 b 移至 c：

move(n−1, a, b, c)

接下来要把（n−1）片金片从 a 经 b 移至 c，我们可以把上面的（n−2）片金片“捆”在一起当作一片金片。此时移动步骤如下：

将 (n−2) 片金片从 a 经 c 移至 b

将 (n−1) 号金片从 a 移至 c

将 (n−2) 片金片从 b 经 a 移至 c

上述过程可描述为：

move(n−2, a, c, b)

n−1 : a → c

move(n−2, b, a, c)

此时已经把（n−1）片金片的移动过程简化为把（n−2）片金片从 a 经 b 移至 c：

move(n−2, a, b, c)

这样如此下去，每次减少一片金片，直到只剩下一片金片为止。

3. 算法描述

这是一种典型的递归问题。递归的边界是无金片可移，即 n=0。其移动过程的算法可描述如下：

move(n, a, b, c)

① if n=0 退出程序

② 调用 move(n−1, a, c, b)

③ 输出“金片号　针号→针号”

④ 调用 move(n−1, b, a, c)

现在很容易把上述描述写成 Java 代码：

```
// 利用递归定义 move( ) 方法
static void move(int n, String a, String b, String c)
{
    if (n!=0)
    {
        move(n − 1, a, c, b);    // 将 (n−1) 片金片从 a 经 c 移至 b
        // 输出移动过程 : 将 n 号金片从 a 移至 c
        System.out.println(n+":  "+a+"  →  "+c);
        k=k+1;                   // 累计移动次数
        move(n − 1, b, a, c);    // 将 (n−1) 片金片从 b 经 a 移至 c
    }
}
```

4. 编程实现

实现上述算法的 Java 程序如下：

```
import java.io.*;
public class Hanoi
{
    static int k=0;
    // 建立 Input( ) 方法，输入一个整数
    static int Input( ) throws IOException
    {
        // 使用缓冲区 (BufferedReader) 从文本数据流读取文本数据
        InputStreamReader reader=new InputStreamReader(System.in);
        BufferedReader input=new BufferedReader(reader);
        System.out.print(" 输入金片数 :");
        String s=input.readLine( );    // 从键盘输入一个字符串
        int n=Integer.parseInt(s);     // 将字符串 s 转换为整数
        return n;
    }
    // 利用递归定义 move( ) 方法
    static void move(int n,String a,String b,String c)
    {
        if (n!=0)
        {
            move(n – 1, a, c, b);      // 将 (n–1) 片金片从 a 经 c 移至 b
            // 输出移动过程 : 将 n 号金片从 a 移至 c
            System.out.println(n+":  "+a+"  →  "+c);
            k=k+1;                     // 累计移动次数
            move(n – 1, b, a, c);// 将 (n–1) 片金片从 b 经 a 移至 c
        }
    }
    public static void main(String args[ ]) throws IOException
    {
        int n = Input( );              // 调用方法输入待移动的金片数
        System.out.println(" 金片号 "+" 针号 "+" → "+" 针号 ");
        move(n,"a","b","c");
        System.out.println(" 移动次数 : "+k);
    }
}
```

当 n=3 时，运行结果如图 3-3-4 所示。

图3-3-4 汉诺塔问题运行结果之一

获得图灵奖的第一位华裔科学家

获得 2000 年图灵奖的美国科学院院士、计算机科学理论学家姚期智博士，是迄今为止获得图灵奖的唯一一位华裔科学家。姚期智博士因在计算理论方面的基础性贡献获得此奖。

姚期智（Andrew C.Yao）祖籍湖北孝感，1946 年圣诞节前夜出生于上海。1967 年，姚期智以优异的成绩毕业于台湾大学，之后赴美深造。在 1972 年取得哈佛大学物理学博士学位以后，姚期智到伊利诺斯大学研究生院继续学习，伊利诺斯大学一向以计算机科学研究领域的深厚积淀而闻名全美。1975 年，姚期智在伊利诺斯大学拿到了他的第二个博士学位——计算机科学博士学位。之后，他曾先后在麻省理工学院（1975-1976）、斯坦福大学（1976-1981，1983-1986）、加州大学伯克利分校（1981-1983）等美国一流高等学府从事教学和研究工作，1986 年至今一直任普林斯顿大学计算机科学系教授。姚期智博士在数据组织、基于复杂性的伪随机数生成理论、密码学、通信复杂性乃至量子通信和计算等多个尖端科研领域，都做出了巨大的、独到的贡献。

1．编写程序，用递归法计算两个正整数的乘积 m×n。

2．求两个自然数的最大公约数的算法最早来自算术中的辗转相除法，人们称之为欧几里得算法。如求自然数 m 和 n 的最大公约数，方法如下：用 m 除以 n，若余数 r 为 0，则此时 n 就是 m 和 n 的最大公约数；若余数 r 不为 0，则将上次的除数 n 作为被除数、余数 r 作为除数，再次相除后求余数。如此不断地反复辗转相除，直到余数 r 为 0 为止，此时，除数就是 m 和 n 的最大公约数。试利用递归算法编程实现。

第四节 排序与查找

批量数据的排序在数据处理中有着极其重要的作用。本节我们将学习对批量数据进行排序与查找，通过学习，掌握选择排序算法以及顺序查找、二分查找等算法。

一 排序及其算法

所谓排序，就是使一串记录，按照其中的某个或某些关键字的大小排列起来的操作，是应用计算机进行数据处理时经常用到的算法。排序的方法有多种，例如，插入排序、冒泡排序、比较排序、选择排序、快速排序等。

1. 问题提出

学校举行体育运动会，需要用计算机处理很多数据，涉及很多排名问题，这里以部分运动员 100 米短跑成绩为例（见表 3-4-1），说明排序的实现方法。

表3-4-1 部分运动员短跑成绩

运动员号码	100米短跑比赛成绩（秒）
30101	12.1
30112	11.4
30201	13.2
30212	11.9
30316	12.8

2. 问题分析

排序的流程如图 3-4-1 所示：

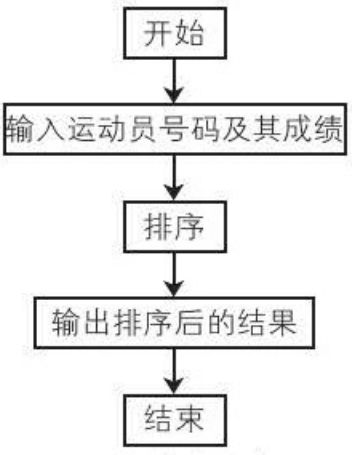


图3-4-1 排序的流程图

这里以比较法为例说明排序算法的思想，我们将五位运动员 100 米的短跑成绩：12.1，11.4，13.2，11.9，12.8 存放在数组 a 中，然后分析数据从小到大（成绩则是由高到低）排序过程中数组元素中数据变化的情况。

12.1	11.4	13.2	11.9	12.8
a[0]	a[1]	a[2]	a[3]	a[4]

首先，取出第 1 个元素 a[0]，逐个地与第 2~5 的元素进行比较，如果有比 a[0] 小的元素，就将 a[0] 与该元素交换。具体过程如下：

①把第 1 个元素 a[0] 与第 2 个元素 a[1] 进行比较，因为 a[0]>a[1]，所以交换 a[0] 与 a[1]。

②把 a[0] 当前的值与第 3 个元素 a[2] 进行比较，因为 a[0]<a[2]，所以不做任何操作。

③把 a[0] 当前的值与第 4 个元素 a[3] 进行比较，因为 a[0]<a[3]，所以不做任何操作。

④把 a[0] 当前的值与第 5 个元素 a[4] 进行比较，因为 a[0]<a[4]，所以不做任何操作。

此时 a[0] 的值为第一轮比较交换之后 5 个数据中的最小值。为了能够使比较过程更清晰地表示出来，我们用表 3-4-2 来表示，其中 i 表示被取出元素的数组下标，j 表示被比较元素的数组下标。每次比较后，如果发生交换，那么交换后的两个数用下划线表示，如果没有发生交换，则仅将本次比较中的较小者用下划线表示。

表3-4-2　第一轮排序过程

i		0	0	0	0
j		1	2	3	4
a[0]	12.1	11.4	11.4	11.4	11.4
a[1]	11.4	12.1	12.1	12.1	12.1
a[2]	13.2	13.2	13.2	13.2	13.2
a[3]	11.9	11.9	11.9	11.9	11.9
a[4]	12.8	12.8	12.8	12.8	12.8

其次，把第 2 个元素 a[1] 逐个地与第 3 个到第 5 个元素进行比较，如果有比 a[1] 小的元素，就将 a[1] 与该元素交换。这样第二轮比较交换之后，a[1] 的值为剩下的 4 个数据中的最小值。

再把第 3 个元素 a[2] 逐个地与第 4 个到第 5 个元素进行比较，如果有比 a[2] 小的元素，就将 a[2] 与该元素交换。这样第三轮比较交换之后，a[2] 的值为剩下的 3 个数据中的最小值。

最后把第 4 个元素 a[3] 与第 5 个元素 a[4] 进行比较，如果 a[3] 大于 a[4]，就将 a[3] 与 a[4] 交换。这样，经过第四轮比较交换之后，a[0] 到 a[4] 的值已按从低到高的顺序排列了。

比较过程中数组 a 中数据变化的情况如表 3-4-3 所示。

表3-4-3　排序全过程

i		0	0	0	0	1	1	1	2	2	3
j		1	2	3	4	2	3	4	3	4	4
a[0]	12.1	11.4	11.4	11.4	11.4	11.4	11.4	11.4	11.4	11.4	11.4
a[1]	11.4	12.1	12.1	12.1	12.1	12.1	11.9	11.9	11.9	11.9	11.9
a[2]	13.2	13.2	13.2	13.2	13.2	13.2	13.2	13.2	12.1	12.1	12.1
a[3]	11.9	11.9	11.9	11.9	11.9	11.9	12.1	12.1	13.2	13.2	12.8
a[4]	12.8	12.8	12.8	12.8	12.8	12.8	12.8	12.8	12.8	12.8	13.2

3. 算法设计

```
for (int i=0; i<a.length-1; i++)
{
   for (int j=i+1; j<a.length; j++)
   {
        比较 a[i] 和 a[j]：当要求排序结果为升序时，如果 a[i]>a[j] 就交换二者；
        当要求排序结果为降序时，如果 a[i]<a[j] 就交换二者。
   }
}
```

4. 编程实现

将上述排序算法定义为一个 Sort( ) 方法。格式为：

```
static void Sort(double a[ ], int s[ ])
```

参数：数组 a[ ] 存放运动员短跑成绩，数组 s[ ] 存放运动员编号。

说明：函数 Sort 执行后，参数数组 a 中的数据按升序排列。

◆ Sort( ) 方法的代码及其说明

```
static void Sort(double a[ ], int s[ ])
{
     double  t;                    // 声明临时变量 t
     int tt;                       // 声明临时变量 tt
     // 比较法排序
     int n=a.length;               // 声明变量 n，存放数组 a[ ] 的长度
     for (int i=0; i<n-1; i++)     // 对数组作 n-1 轮比较
     {
          for (int j=i+1; j<n; j++)  // 每一轮依次与后面的数组元素进行比较
          if (a[i]>a[j])             // 如果 a[i]>a[j]，那么
          {
               t=a[i]; a[i]=a[j]; a[j]=t;     // 交换 a[i] 和 a[j]
               tt=s[i]; s[i]=s[j]; s[j]=tt;   // 交换 s[i] 和 s[j]
          }
     }
}
```

◆输出结果模块

```
方法名：printArray( )
参数：数组 s[ ]，a[ ]
返回值：字符串 result
```

输出结果模块代码如下：

```
// 定义 printArray( ) 方法，输出数组 s[ ],a[ ]
static String printArray(double a[ ], int s[ ])
{
        String result="";        // 声明并初始化字符串对象 result
```

```
        int n=a.length;          // 计算数组 a[ ] 的长度
        for (int i=0; i<n; i++)
            result+=s[i]+":"+a[i]+" ";
            // 将运动员的编号与成绩对应连起来
        return result;
}
```

◆ main( ) 方法

```
public static void main(String args[ ])
{
        // 声明并初始化数组 s[ ]，s[ ] 存放运动员编号
        int s[ ]={30101,30112,30201,30212,30316};
        // 声明并初始化数组 a[ ]，a[ ] 存放 100 米成绩
        double a[ ]={12.1,11.4,13.2,11.9,12.8};
        // 输出处理前的数组 a[ ]、s[ ]
        System.out.println(" 排序前 ：\n"+printArray(a,s));
        // 调用排序方法 Sort( )
        Sort(a,s);
        // 输出排序后的数组 a[ ]、s[ ]
        System.out.println(" 排序后 ：\n"+printArray(a,s));
}
```

5. 程序优化

比较排序法是一种较易理解的排序算法，但是在排序过程中交换的次数较多，排序的效率低。现在我们对其进行改进，在 a[i] 与 a[i+1] 到 a[n] 都比较完后，将 a[i] 与 a[i+1] 到 a[n] 中最小值的那个元素交换。这样每一轮比较之后，只需交换一次。经过 n−1 遍之后，就可以完成 n 个数据的排序操作。这样改进以后的排序方法叫做选择排序法。

（1）选择排序算法分析

①把 n 个数据存放在一维整数数组中。

②先取出第 1 个元素 a[0]，逐个地与第 2 到第 n 个元素进行比较，如果有比 a[0] 小（大）的，就记下该元素的位置 k。比较完一遍之后，将 a[0] 与 a[1] 到 a[n−1] 中值最小（大）的那个元素交换，结果 a[0] 必然是最小（大）的。

③再取出第 2 个元素 a[1]，逐个与第 3 到第 n 个元素进行比较，如果有比 a[1] 小（大）的，就记下该元素的位置 k。比较完一遍之后，将 a[1] 与 a[2] 到 a[n−1] 中值最小（大）的那个元素交换，这样，a[1] 必然是除了 a[0] 以外最小（大）的。

这样每一轮比较之后，只需交换一次。经过 n−1 遍之后，就可以完成 n 个数据排序的操作。

（2）算法描述

```
for (int i=0; i<a.length−1; i++)
{
    从 [ i+1,n ] 区间内找最小值元素的位置 ( 下标 )k;     // n= a.length
```

```
    将下标为 k 的元素与下标为 i 的元素进行交换；
}
```

（3）代码实现

```
// 选择法排序
static void Sort2(double a[ ], int s[ ])
{
    double t;          // 声明临时变量 t
    int tt;            // 声明临时变量 tt
        int n=a.length;              // 声明变量 n，存放数组 a[ ] 的长度
      for (int i=0; i<n-1; i++)      // 对 n 个元素的数组作 n-1 轮比较
      {
            int k=i;                 // 记下 i 位置
            for (int j=i+1; j<n; j++)  // 每一轮依次与后面的数组元素进行比较
            if (a[k]>a[j])     k=j;    // 如果 a[k]>a[j]，那么记下 j 值
            if (k!=i)                  // 如果 k ≠ i，那么
            {
                t=a[k]; a[k]=a[i]; a[i]=t;      // 交换 a[k]、a[i]
               tt=s[k]; s[k]=s[i]; s[i]=tt;     // 交换 s[k]、s[i]
            }
      }
}
```

二 查找及其算法

查找又称检索，是数据处理中经常用到的一种操作。查找就是从一个数据集合中找出特定元素的过程。数据集合中的元素一般应是同类型的对象。

查找通常分为顺序查找和二分查找两种。下面仍以表 3-4-1 中的数据为例，说明查找的算法及其实现。

例如，学校春季运动会 100 米短跑项目比赛结束后，需要查找某运动员的成绩。程序流程图如图 3-4-2 所示。

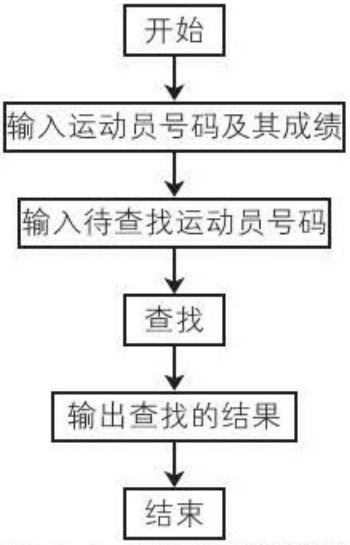


图3-4-2　查找程序的流程图

1. 顺序查找法

（1）算法描述

顺序查找又称为线性查找，它的基本思想是对所存储的数据从第一项开始，依次与所要查找的数据进行比较，直到找到该数据或将全部数据比较完为止。

（2）代码实现

```
static int linearSearch(int a[ ], int key)
{
```

```
        // 逐个元素与 key 相比较，找到则返回 i 值
        for (int i=0; i<a.length; i++)
            if (a[i] == key) return i;
        return -1;                  // 否则返回 -1
    }
```

（3）程序实现

调用 linearSearch( ) 方法的完整程序代码如下：

```
// 顺序查找
import javax.swing.JOptionPane;   // 导入所需要的 JOptionPane 类
public class Search1
{
    static int linearSearch(int a[ ], int key)
    {
        // 逐个元素与 key 相比较，找到则返回 i 值
        for (int i=0; i<a.length; i++)
        if (a[i] == key) return i;
        return -1;                  // 否则返回 -1
    }

    public static void main(String args[ ])
    }
         int a[ ]={30101,30112,30201,30212,30316};
        double b[ ]={12.1,11.4,13.2,11.9,12.8};
        // 使用组件 JOptionPane 的输入对话框输入文本数据
        String s=JOptionPane.showInputDialog(" 请输入待查找运动员号码 :");
         int key=Integer.parseInt(s);          // 将字符串 s 转换为整数
         int k = linearSearch(a, key);         // 调用顺序查找 linearSearch( ) 方法
         // 调用 JOptionPane 的消息对话框，显示查找结果
         if (k != -1)
         JOptionPane.showMessageDialog(null, " 运动员成绩是 : " + b[k],
         " 查找运动员 " + key + " 的成绩 ",JOptionPane.PLAIN_MESSAGE);
         else
         JOptionPane.showMessageDialog(null," 运动员成绩未找到 !",
         " 查找运动员 " + key + " 的成绩 ",JOptionPane.PLAIN_MESSAGE);
    }
}
```

程序经编译运行后，首先出现一个输入对话框，如图 3–4–3 所示。

在输入待查找的运动员号码后，单击“确定”按钮，即可显示查找结果，如图 3–4–4 所示。

图3-4-3 输入对话框

3-4-4 显示查找结果

知识拓展 JoptionPane类的常用方法

程序中可以利用 JoptionPane 类的组件来设计标准的对话框，用于显示提示信息。这需要在程序代码中导入所需要的 JOptionPane 类：

```
import javax.swing.JOptionPane;
```

JoptionPane 类中常用方法的调用如下：

①建立输入对话框窗口的方法 showInputDialog( )，例如：

```
String s=JOptionPane.showInputDialog(" 请输入待查找运动员号 :");
```

表示建立一个输入对话框窗口，显示输入提示信息“请输入待查找运动员号 :”，等待输入字符串。单击“确定”按钮后，将输入的字符串赋值给 s。

②显示提示信息的方法 showMessageDialog( )：

```
public static void showMessageDialog(Component  parentComponent,
                Object message,String title,int messageType)
```

它的作用是在父组件之下建立一个消息对话框窗口，显示提示信息字符串 message。JoptionPane 的父类是 Jcomponent；title 为窗口标题字符串；messageType 为消息对话框的类型，常用的有五种类型，见表 3-4-4。

表3-4-4 消息对话框类型

messageType	含 义
JOptionPane.INFOTMATION_MESSAGE	信息对话框
JOptionPane.ERROR_MESSAGE	错误对话框
JOptionPane.WARNING_MESSAGE	警告对话框
JOptionPane.QUESTION_MESSAGE	问题对话框
JOptionPane.PLAIN_MESSAGE	普通对话框

2. 二分查找法

在存储几万名考生成绩的数据集合中，要根据准考证号查找某考生各科的成绩，可使用二分法。各考生的成绩若是按准考证号从小到大或从大到小的次序存放的，要查找某考生的成绩就较容易，否则只能逐个考察各考生的准考证号，这样查找是很费时的。

为了提高从数据集合中查找特定元素的效率，首先要对集合中的数据进行排序。即把各元素按关键字从小到大或从大到小进行排列，然后存储在一维数组中。这样就可以使用查找效率较高的二分查找法了。

二分查找的过程是先确定待查找元素所在的范围，然后逐步缩小范围，直到找到或发

现不存在该元素为止。

每次查找都把查找的区间一分为二，把待查找的数据和中间的元素进行比较。如果二者相等，则表明找到了要找的元素，即输出该元素的下标；如果数据大于中间元素，则到比中间元素值大的那一半区间去查找，否则到比中间元素值小的那一半区间去查找。这样经过一次比较可以将查找的区间减小一半。这是一种应用分治策略的解题思想。

二分查找法的算法如下：

```
while ( 区间低端值不大于高端值 )
{
    计算区间的中点下标值 ;
    if ( 中点处的元素值等于指定值 ) { 返回区间中点的下标 }
    else { 确定下一步考察的区间 }
}
```

编写程序时，可以用 left 表示区间最小元素的下标，用 middle 表示区间中间元素的下标，用 right 表示区间最大元素的下标。每次比较两个数，若不等，就把 middle 的值赋给 left 或 right。

二分法 binarySearch( ) 代码段如下：

```
// 二分查找
static int binarySearch(int a[ ], int key)
{
    int left = 0;
    int right = a.length – 1;
    int middle;
    while (left <= right)
    {
        middle = (left + right) / 2;
        if (key == a[middle])
            return middle;
        else
            if (key < a[middle])
                right = middle – 1;
            else
                left = middle + 1;
    }
    return –1;
}
```

完善“二分查找法”。

编程实现：随机产生 1 000 个一万以内的整数，任意查找某数是否存在，并输出其所在的位置。

到目前为止，我们已经学习了解析法、穷举法、递归法以及排序查找等基本算法，并利用这些算法解决了一些实际问题。请尝试归纳出算法与待解决的问题之间的关系，讨论使用程序设计解决问题的优势与局限性，并上网查找更多有关算法的知识，例如利用递归实现的回溯算法等。了解这些算法后，可用其编程，解决实际问题。

机器证明

机器证明是用电子计算机来完成数学命题的证明，它是现代数学中一个新兴的边缘学科，是现代人工智能发展的一个重要方向。利用计算机进行自动推理，特别是进行几何定理的自动证明，是学术界长期研究的课题。早在17世纪，数学家莱布尼兹就提出过用机器证明的设想。直到19世纪末，数学家希尔伯特从公理化思想出发，提出了数理逻辑，这才使机器证明问题具有了明确的数学形式。

1950年，波兰数理科学家塔尔斯基进一步从理论上证明，初等代数和初等几何的定理可以实行机械化推理。1956年，美国卡内基大学—兰德公司协作组在电子计算机上成功地证明了罗素和怀特海所著的《数学原理》第二章52条定理中的38条。这是机器证明的开始。

1959年，美籍华人王浩教授只用9分钟时间，就在计算机上证明了罗素和怀特海《数学原理》一书中一阶逻辑部分的全部定理350多条，引起数学界的轰动。

1965年美国数学家鲁滨逊提出了著名的归结原理，该原理的基本出发点是，要证明任何一个命题为真，都可以通过证明其否定为假来得到。

1976年，美国数学家阿佩尔和黑肯借助于计算机，成功地证明了手工证明所没有解决的重要难题——“四色猜想”。

1977年，我国数学家吴文俊（中国科学院院士）确认了初等几何一类主要定理的证明可以机械化。1980年，他用一台微机进行几何问题的推断，在20小时内发现了一个几何学的新定理，又用60小时左右的时间发现了另一个几何学的新定理，这引起了学术界的关注。从手工证明到机器证明，是数学思想方法的重大飞跃。国际上把吴文俊提出的关于“数学机械化”的研究成果称为“吴方法”。为此，吴文俊院士获得2000年度中国国家最高科技奖。

1．编程让计算机自动产生20个随机两位整数，并按由大到小的顺序排列。

2．排序有很多算法，除了前面介绍过的选择排序法之外，还有冒泡法排序、插入法排序、快速排序等。它们排序的效率不一样。

在安装JDK系统的文件夹中，有几个排序算法的演示。演示文件为“jdk\demo\applets\SortDemo”路径下的“example1.html”。请打开该文件，观看演示效果，并打开Java的源程序，了解、比较有关的排序算法。也可以从网上查找、下载排序与查找算法的其他相关源程序，并上机实践。

第四单元 · 尝试软件开发

面对生活中的实际问题，同学们是否都能找到有效的解决办法？对于复杂的问题应当怎样解决？是不是先将其分解成若干个简单的问题后再逐个解决的？分解解决其实就是“分而治之”的解题思路。设计一个成熟的软件系统，需要根据分解的子问题研究相应的算法，编制相应的程序模块，最后将它们有机地集成起来。此外，软件的开发还要用规范的思想进行指导。

“千里之行，始于足下。”一开始开发软件，我们会因为生疏而有些胆怯，但熟能生巧，只要坚持不懈，努力积累编程经验，我们就一定能成为一名出色的编程“高手”。

第一节 项目策划

我们以一个主题网站建设为例，进行开发软件项目的尝试，并用规范的思想指导软件开发的全过程。经过这一项目的尝试，最终学会有效解决复杂的实际问题。

在学习了算法设计以及如何进行程序设计的基本知识以后，我们要编写一个小程序是不成问题的。但是要开发一个综合性的软件系统，还会遇到很多困难。什么是软件系统呢？暂时可以狭义地把它理解为大型程序。软件开发不同于编写小程序，自从“软件危机”出现以后，人们就认识到应该将软件开发的全过程视为一项工程来处理。软件工程的目的是以较低的成本开发出较高质量的软件。实质上，它是一套规范的管理模式。因此，要学会使用规范的思想来指导软件的开发。

我们将以开发主题网站作为软件开发的项目，以分小组完成任务的形式，贯穿本单元的学习。

用规范的思想指导软件项目的开发，需要把软件项目的开发过程分为六个主要环节，包括项目管理、需求分析、系统设计、程序设计、测试与维护等。其中，需求分析与系统设计属于项目策划阶段，程序设计、测试与维护属于项目实现阶段。

一 项目管理

项目管理作为重要环节，将贯穿整个开发过程。

1. 项目立项

随着因特网的发展，网站作为一种新的应用平台，被越来越多的人所重视。我们确定的项目是开发一个主题网站。至于是什么主题，各小组可根据兴趣、爱好或实际需求来讨论确定。现在我们以开发一个宣传奥林匹克运动会的主题网站为例，暂且命名为“北京冬季奥运会 2022”。

2. 成立项目小组

根据立项主题，小组成员应包括项目经理、设计员、程序员、测试员、文档编辑员等。项目实行项目经理制。每种角色的人数可以视项目具体情况确定，一个人也可以担当几种角色。

现在请大家把分组情况填入表 4-1-1 中。

表4-1-1　项目小组成员分工表

组　名	项目经理	设计员	程序员	测试员	文档编辑
第一组					
第二组					
第三组					
第四组					
第五组					
……					

3. 制定项目规划

制定项目规划就是要确定工作进程，填写项目规划书，让所有人员明确任务、步调一致，最终准时完成任务。项目规划书包括项目名称、各阶段的日程安排、时间进度、资源规划等。可参考表 4-1-2 样式填写。

表4-1-2　项目规划书

项目名称				
阶段名称	时间安排	具体内容		备　注
可用的资源 人员				
可用的资源 软件				
可用的资源 硬件				
文档记录	编制	审核	确认	日期

资源规划包括人力资源和可用的软、硬件资源。网站建设可以使用很多开发工具，我们这里以所学过的主页制作工具和 Java 程序设计语言作为主要的开发工具。

4. 质量检查

在软件开发过程中，要定期进行质量检查，质量检查应遵循以下几条原则：

（1）事先列出要检查的主要内容。

（2）只评审工作，说清楚问题所在，不评审开发者。评审的气氛应该是融洽的。应当有礼貌地指出存在的错误，要重视每个人的意见。

（3）建立一个议事日程并严格遵照执行。检查过程不能放任自流，必须按照既定的方向和日程进行检查。

（4）不要花太多的时间争论和辩驳。

二 需求分析

1. 提出需求

需求方应提出一份完整的需求说明。我们以校园网站的开发为例。

教育部对校园网的定义是：校园网是为学校师生提供教学、科研和综合信息服务的宽带多媒体网络。首先，校园网应为学校教学、科研提供先进的信息化教学环境；其次，校园网应具有教务、行政和总务管理功能；最后，校园网还应满足校内外的通信要求。

学校网站是实现教育资源分配的桥梁。网络有巨大的教育资源库，它集全社会的力量，使教育资源无限增长。过去，发达地区和欠发达地区，高投入学校和条件差的学校在获取教育资源的权力上很难平等，每一位教师和学生得到培训和受教育的机会也不平等。有了网络教育资源库，学校水平、教材、教师能力的影响大为缩小，这不仅能极大地提高教学效率，而且能实现教育公平的社会理想。

学校网站能提供教学互动的全新方式，使得教师与教师、教师与学生、学生与学生之间的交流有了全新的方式。它不再受传统课堂的制约，它可以使身处异地的学校如同同处一室，共同讨论，共同共享，它使地理上的界限模糊甚至消失了。学校网站是真正没有围墙的学校。

学校网站能够提供个性化的学习平台。不同的学生理解世界的方式各不相同，认知世界方式也有诸多不同。网络提供的丰富资源可以使学生寻找适合自己的学习方式，各取所需。学校网站允许学生沿着自己的方向，按照自己的速度进行学习和接受教育。学生有机会享受最佳的教育机会，充分发掘自己的内在潜力，培植独特的个性和人格。

学校网站是学校的“商标”，每一所学校都有自己的特色，每一所学校都有自己的个性。在这个高度信息化的社会里，建立学校网站是最直接的宣传手段。网站没有区域限制的特性，不仅能让当地人了解学校，也能让外地人甚至全世界的人了解学校。凭借学校网站，学校就可成为教育百花园中一朵鲜花。

对于校园网站，学校领导、教师、学生、家长、社会都会有各自的需求。所处的位置不同，看问题的角度不同，需求也就不同，但大家的目标是一样的，就是使学生成才。作为学生的你有什么需求呢？

学生的需求有很多，比如有的同学提出可以在网上上课，在网上提问交流，网上交作业，网上测试；有的同学想通过网络阅读了解课外知识；有的同学提出建立个人主页，班级空间，发表自己的博客，拥有自己的网络硬盘；还有的同学希望通过网站了解班级、学校发布的信息和班级、学校组织的各项活动等等。

总之，你想利用该主题网站表现什么内容，一定要列出一份详细的需求说明。

2. 编制需求分析报告

编制需求分析报告应遵循以下几点原则：

- 正确性：必须正确地描述清楚每项需求的功能。
- 可行性：确保在当前的开发能力和系统环境下可以实现每项需求。
- 必要性：是否必须同时交付功能齐全的需求系统，有些功能是否可以推迟实现。
- 简明性：尽量不使用过于专业的术语。
- 检测性：开发完毕后，可以根据此需求报告进行检测。

需求分析报告一般用文档编辑软件来完成。

三　系统设计

系统设计可分为总体设计和详细设计两个阶段，设计的最后要形成一个系统设计方案，作为后期项目实现的依据。因此，系统设计是非常关键的一步，是把用户需求转化为软件系统的最重要的环节。

1. 总体设计

总体设计主要涉及以下五个方面的内容。

（1）网站需要实现的功能

可从需求分析报告中提炼出来。

（2）网站开发使用的软件、硬件环境

我们在信息技术的必修课部分已经学过了一些主页制作软件，在本书前面又学过了Java 程序设计，因此我们可以使用 Dreamweaver 制作主页页面，使用绘图软件如 Photoshop 处理图形与图像，使用 Java 进行网页特效设计及小程序的处理。可以先在单机上开发，最后上传到服务器中，利用 WWW 服务发布主页。

（3）需要的人员和最后完成期限

视班级分组情况和项目规模而定。

（4）需要遵循的规则和标准

制定统一的编程规范，统一网站的结构。必须按照约定的原则为文件、目录命名，这样便于最后合成。要把大的项目划分成若干个模块，这就是“分而治之”的编程思想。各模块尽量做到既具封闭性又具开放性，即模块可以作为一个独立的整体而被其他程序调用（封闭性），同时这个模块又可以扩充（开放性）。只有遵循统一的编程规范，才能保证代码的开放性。采用面向对象的设计方法可以较好地解决这个问题。

（5）编写总体规划报告

总体规划报告的内容包括：

- 网站的栏目和板块；
- 网站的功能和相应的程序；
- 网站的链接结构；
- 如果有数据库，进行数据库的设计；
- 网站和用户的交互性设计。

2. 详细设计

总体设计阶段以比较抽象概括的方式提出解决问题的办法，详细设计阶段的任务就是把解决方法具体化。详细设计主要是针对程序开发部分说的，但这个阶段不是真正编写程序，而是设计出程序的详细规格说明。规格说明的作用类似于其他工程领域中工程师经常使用的工程蓝图。规格说明应该包含必要的细节，如程序界面、表单、需要的数据等等。程序员可以根据它写出实际的程序代码。

在程序员进行详细设计的同时，网页设计师开始设计网站的整体形象和首页。

整体形象设计包括网站标志、标准色彩、广告语等。

首页设计包括版面、色彩、图像、图标以及动态效果等风格设计，也包括菜单、标题、版权等模块设计。首页一般设计 1 ~ 3 个不同风格，供小组成员讨论选择，确认后最好不再对版面风格作大的变动。

在作完总体设计和详细设计后，一定要编制出系统设计方案。请参考图 4-1-1 所示的系统设计方案目录，编写一套完整的“校园网”网站的系统设计方案。

>>> 系统设计方案<<<

目 录

1. 客户情况分析
2. 网站需要实现的目的和目标
3. 网站形象说明
4. 网站的栏目板块和结构
5. 网站内容的安排，相互链接关系
6. 所用软件、硬件和技术的分析说明
7. 开发时间进度表
8. 宣传推广方案
9. 维护方案
10. 制作成本
11. 项目开发小组简介

第二节　项目实现

在这一阶段，小组成员要紧密团结，发扬协作精神，共同完成项目的开发。最后，各组要进行作品展示，通过评价与交流，取他人所长，补自己之短，使每个人的信息素养都得到提升。

在这一阶段，程序员、网页设计人员和测试人员要按照分工同时开展工作。项目经理要经常了解项目进度，协调和沟通程序员与网页设计人员的工作。

一　程序开发

1. “分而治之”策略

“分而治之”就是将一个复杂的问题分解成若干个简单的问题，然后逐个解决。因此我们根据网站功能及栏目设置情况，分栏目、分模块开发相应的程序。分解后的每个问题都必须能通过编程来求解，而且所有程序最终要能够集成为一个能解决原始问题的大程序。这也是模块程序设计的思想。

例如，要在“校园网”网站设计一个“学生最喜欢的个人主页评选”的投票程序。

我们可以把这个问题分解为几个小问题，如：

◆ 输入模块：选择一个个人主页的编号；

◆ 累加模块：根据输入的数值，给对应的主页计数器加 1；

◆ 输出模块：显示每个参评主页的累计数值，查找出最大的数值，输出其对应的主页为“学生最喜欢的主页”。

我们曾在第二单元第三节实现了类似该程序的输入子模块，大家可以用所学过的知识，继续完成其他子模块的编写。因为最终要放到网站上运行，所以应该编写 Applet 小程序加以实现。

2. 代码重用策略

程序重用就是利用现成的程序进行程序设计。将具有一定集成度并可以重复使用的程序或软件组成单元，称为构件。构造一个软件系统不必每次从零做起，可以直接使用已有的构件，即可组装或经合理修改后成为新的系统。

代码重用不仅要使自己使用方便，还要让别人使用也方便，这就是面向对象程序设计的思想。

可重用的代码很多，因特网上提供了大量的 Java 免费源代码，能实现各种网页特效或一些专门的功能。上网搜一搜，下载后做适当的修改，生成自己需要的程序，体验一下代码重用的乐趣吧！

例 1：运行如下的校庆倒计时程序代码，程序运行结果如图 4–2–1 所示。

图4–2–1 校庆倒计时

下面程序中校庆时间为 2010 年 10 月 17 日，GregorianCalendar dayAoyun = new Gregorian–Calendar(2010,9,17); 语句中的 2010,9,17 对应校庆时间，注意 0 表示 1 月，以此类推，所以 9 表示 10 月。

我们可以自己修改此程序代码的标题和时间，编写适合自己的倒计时程序。

```
import java.awt.*;
import javax.swing.*;
import java.util.*;
public class timer extends JFrame{
   public timer( ) {
      super(" 校庆倒计时 ");
      setBackground(Color.WHITE);
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      setBounds(150,150,300,80);
      JLabel label1 = new JLabel(" 距校庆还有 ： ");
      JLabel label2 = new JLabel(getDay( ));
      label2.setForeground(Color.RED);
      label2.setFont(new Font(" 黑体 ",Font.BOLD,20));
      JLabel label3 = new JLabel(" 天！  ");
      Container pane = getContentPane( );
      pane.setLayout(new FlowLayout( ));
      pane.add(label1);
      pane.add(label2);
      pane.add(label3);
      this.setVisible(true);
   }
   public String getDay( ){
    GregorianCalendar now = new GregorianCalendar( );
      // 倒计时日期 2010 年 10 月 17 日，注意月份 0 表示 1 月，以此类推
      GregorianCalendar dayAoyun = new GregorianCalendar(2010,9,17);
      long temp = dayAoyun.getTimeInMillis( ) – now.getTimeInMillis( );
      long days = temp / (1000 * 60 * 60 * 24) + 1;
      String day = String.valueOf(days);
      return day;
   }
```

```
    public static void main(String[] args){
            timer  time1 = new timer( );
    }
}
```

例2：教科书配套光盘目录\sc\unit4\clock中有4个文件，分别是anclock.class，anclock.jar，clock.jpg和clock.html。其中，anclock.class，anclock.jar是Java的特效代码，功能是显示时钟。这也是可以重用的程序代码的一个例子。clock.jpg是钟表的表盘图；clock.html是网页文件，它通过调用Java程序实现时钟的显示。clock.html的运行结果如图4-2-2所示。

图4-2-2 “clock.html”网页

clock.html网页的源代码如下：

```
<HTML><HEAD><TITLE>Anfy Clock applet</TITLE></HEAD>
<BODY TEXT="#CCFFFF"><p><br></p><center><p>
<APPLET ARCHIVE="anclock.jar" CODE="anclock.class" WIDTH="100"
HEIGHT="100">
<PARAM NAME="credits" VALUE="Applet by Anfy Team (www.anfyteam.com)">
<PARAM NAME="regcode" VALUE="no">
<PARAM NAME="reglink" VALUE="no">
<PARAM NAME="regnewframe" VALUE="yes">
<PARAM NAME="regframename" VALUE="_blank">
<PARAM NAME="statusmsg" VALUE="Go to AnfyTeam">
<PARAM NAME="mode" VALUE="old">
<PARAM NAME="backimg" VALUE="clock.jpg">
<PARAM NAME="startx" VALUE="0">
<PARAM NAME="starty" VALUE="0">
<PARAM NAME="centerx" VALUE="yes">
<PARAM NAME="centery" VALUE="yes">
<PARAM NAME="fgcolor" VALUE="eebb33">
<PARAM NAME="bgcolor" VALUE="001166">
<PARAM NAME="shadow" VALUE="yes">
```

```
<PARAM NAME="shadowcolor" VALUE="000000">
<PARAM NAME="oldc_radius" VALUE="43">
<PARAM NAME="oldc_ticks" VALUE="no">
<PARAM NAME="memdelay" VALUE="1000">
<PARAM NAME="priority" VALUE="3">
<PARAM NAME="MinSYNC" VALUE="10">
Please <A HREF="http://www.anfyteam.com/java/">download Java(tm)</A>.
</APPLET>
<p align=center>
<p></P>
</BODY></HTML>
```

其中，<APPLET> 与 </APPLET> 中间的语句表示嵌入了 anclock.class 小程序，<PARAM NAME=" " VALUE=" "> 之类的语句是参数语句，是 anclock.class 提供的可供修改的参数接口。

例如：语句 <PARAM NAME="backimg" VALUE="clock.jpg"> 表示表盘的图片文件是 clock.jpg。如果我们自己制作一幅图 clock1.jpg，存放到同一目录下，并把该语句中的 clock.jpg 改为 clock1.jpg，运行后就会改变界面，如图 4-2-3 所示。

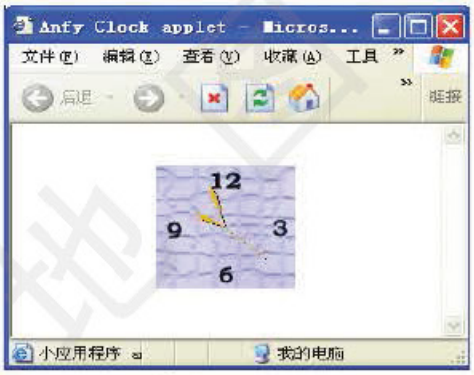


图4-2-3 修改后的clock.html网页

请大家试着修改其他一些参数语句，看看会有什么变化。如果能说出上述源代码中那些参数语句的功能，那就更棒了！

二 测试并完善

测试人员要随时测试网页与程序的运行情况，发现漏洞要立刻记录并反馈给有关人员进行修改，不要等到全部制作完毕再测试，那样会浪费大量的时间和精力。在网站初步建成后，要上传到服务器进行整个范围的测试，包括速度、兼容性、交互性、链接的正确性、程序的健壮性以及超流量测试等，发现问题要及时记录下来并加以解决。

从项目策划到项目调试，我们一直在不停地制定文档、编写文档、记录文档。其实，基于规范思想开发的软件项目本身就是一个文档，是一个不断充实和完善的标准。通过不断地发现问题，解决问题，修改和补充文档，可使这个标准越来越规范，越来越标准化。这样才能使得网站开发趋向规范，趋向合理。

三 作品评价

至此，我们的项目开发暂告一段落，下面各组开始对本组作品进行展示与评价，并进行口头汇报。在各组相互交流的基础上，个人要把学习程序设计及软件开发的所得记录到电子学习档案袋“我的感受”中，并以“定稿”的方式发布。

请大家制定一个统一的评价量规表，评价指标要全面、精练、可行。可根据本班实际情况对表 4-2-1 所示的评价量规表进行修订。在开始评价前，各组还可以对作品进行修改。评价不是目的，只是促进大家提高能力的手段。

表4-2-1 评价量规示意表

分 类	评价指标	评分标准				
		1分	2分	3分	4分	5分
主 题	（1）主题是否明确					
	（2）内容是否全面，能否包括项目要求的所有主题及其他相关主题					
	（3）主题内容逻辑顺序是否清晰					
	（4）重点是否突出					
	（5）主题的表达能否引发思考和探寻更多信息的动机					
作 品	（6）作品表现是否富有新意					
	（7）能否准确合理地应用声音、动画、视频等表达主题					
	（8）整体布局是否均衡合理					
	（9）页面设计是否与主题风格一致					
	（10）是否美观并具有一定的艺术性					
	（11）图片、动画的使用是否合理并有助于理解相关文本					
	（12）是否提供了用于导航和检索的目录页					
	（13）是否具有友好的人机交互界面					
	（14）链接是否准确					
	（15）是否体现了技术创新和创作性					
演 讲	（16）语言是否准确、生动					
	（17）表达是否条理清晰、易于理解					
	（18）能否根据听众的特点灵活地使用信息传递和交流技巧					
协 作	（19）分工是否明确					
	（20）小组成员能否在完成各自任务的同时相互合作，共同完成任务					
总体评价		合计得分：				

最后，请大家打开自己的电子学习档案袋，回顾一下本学期的学习过程，看看自己的知识是不是有了增长，程序设计水平是不是有了提高，开发软件的能力是不是具备了。希望每个人的答案都是肯定的！

重要术语中英文对照表

	中文	英文
B	包	package
	编译	compile
	变量	variable
	表达式	expression
	布局	layout
C	查找	search
	常量	constant
	程序	program
	程序设计	program design
	程序设计语言	program design language
D	递归	recursion
	对象	object
	多态	polymorphism
F	方法	method
	封装	encapsulation
	赋值	assignment
G	构造方法	constructor
H	汇编语言	assembler language
J	Java 小应用程序	Java applet
	Java 虚拟机	Java Virtual Machine（JVM）
	机器码	machine code
	机器语言	machine language
	继承	inheritance
	接口	interface
	解释	interpret
	进程	process
L	流程图	flow chart
	路径	path
M	面向对象的程序设计	Object Oriented Programming（OOP）
P	排序	sort
R	容器	container
S	实例	instance
	实现	implementation
	事件	event
	属性	attribute
	数组	array
	素数	prime number

重要术语中英文对照表

	中文	英文
S	算法	algorithm
W	伪代码	pseudo-code
X	下标	subscript
	线程	thread
Y	应用程序	application
	优先	precedence
	源代码	source code
	运算符	operator
Z	重载	overload
	注释	comment
	字符	character
	字节	byte
	自然语言	natural language
	组件	component

责任编辑　沈万君　兰大鹏
封面设计　张　萌

ISBN 978-7-5031-5372-3

书　　号　ISBN 978-7-5031-5372-3
批 准 文 号

定价：　　元
（含光盘定价：5.00元）