目　录

出版者的话
专家指导委员会
译者序
译者简介
前言
第1章　计算机、Internet和Web ……1
1.1　简介……2
1.2　什么是计算机……5
1.3　计算机的组成……6
1.4　操作系统的演化……6
1.5　个人计算、分布式计算与客户／服务器计算……7
1.6　机器语言、汇编语言和高级语言……7
1.7　C++的历史……8
1.8　Java的历史……9
1.9　Java的类库……10
1.10　其他高级语言……11
1.11　结构化编程……11
1.12　Internet和WWW……12
1.13　典型Java环境的组成……12
1.14　关于Java和本书的通用注释……15
1.15　关于对象的思考：对象技术和统一建模语言简介……17
1.16　揭示设计模式：简介……20
1.17　本书概况……22
1.18　（选学）纵览利用UML进行面向对象设计的实例研究……32
1.19　（选学）浏览“揭示设计模式”的各个小节……35
第2章　Java应用程序……45
2.1　简介……46
2.2　第一个Java程序：打印一行文本……46
2.3　修改第一个Java程序……50
2.3.1　用多行语句显示一行文本……50
2.3.2　用一行语句显示多行文本……51
2.4　在对话框中显示文本……52
2.5　另一个Java应用程序：整数相加……55
2.6　存储器概念……60
2.7　算术运算……61
2.8　判断：相等和关系操作符……63
2.9　（可选实例研究）关于对象的思考：考察问题陈述……69
第3章　Java applet……87
3.1　简介……87
3.2　Java 2软件开发工具箱提供的简单applet示例……88
3.2.1　TicTacToe applet……88
3.2.2　DrawTest applet……91
3.2.3　Java2D applet……91
3.3　一个简单的Java applet：显示一个字符串……92
3.4　两个简单的applet：画字符串和线条……98
3.5　另一个Java applet：浮点数相加……100
3.6　在Web浏览器中查看applet……107
3.6.1　在Netscape Navigator 6 中查看applet……107
3.6.2　使用Java插件在其他浏览器中查看applet……107
3.7　Internet和WWW上的Java applet资源……109
3.8　（可选实例研究）关于对象的思考：标识问题陈述中的类……111
第4章　控制结构（第1部分）……123
4.1　简介……123
4.2　算法……124
4.3　伪代码……124
4.4　控制结构……124
4.5　if选择结构……126
4.6　if／else选择结构……127
4.7　while循环结构……131
4.8　算法设计：实例研究1（计数器控制的循环）……132
4.9　自上而下逐步细化地构成算法：实例研究2（标记控制的循环）……136
4.10　自上而下逐步细化地构成算法：

实例研究3（嵌套的控制结构）……143
4.11 赋值操作符 ……147
4.12 增量和减量操作符 ……148
4.13 基本数据类型 ……150
4.14 （可选实例研究）关于对象的思考：
标识类属性 ……151
第5章 控制结构（第2部分）……165
5.1 简介……165
5.2 由计数器控制的循环……166
5.3 for循环结构 ……167
5.4 for结构应用举例 ……171
5.5 switch多重选择结构 ……175
5.6 do/while循环结构……179
5.7 break和continue语句 ……181
5.8 带标号的break和continue语句 ……184
5.9 逻辑操作符……185
5.10 结构化编程小结 ……191
5.11 （可选实例研究）关于对象的思考：
标识对象的状态和活动 ……195
第6章 方法 ……207
6.1 简介……208
6.2 Java中的程序模块 ……208
6.3 Math类的方法 ……209
6.4 方法……209
6.5 方法定义……210
6.6 数据类型的提升……216
6.7 Java API包 ……217
6.8 随机数的生成……218
6.9 例子：碰运气游戏……222
6.10 标识符的持续时间 ……229
6.11 作用域规则 ……230
6.12 递归 ……232
6.13 递归举例：Fibonacci数列 ……235
6.14 递归与迭代 ……239
6.15 方法重载 ……241
6.16 JApplet类的方法 ……243
6.17 （可选实例研究）关于对象的思考：
标识类操作 ……244
第7章 数组 ……263
7.1 简介 ……263
7.2 数组……264
7.3 声明数组和给数组分配内存……265
7.4 数组应用举例……266
7.4.1 分配数组内存并初始化数组元素 …266
7.4.2 用初值表对数组进行初始化 ……267
7.4.3 计算并存储数组元素的值 ……268
7.4.4 计算数组元素之和 ……270
7.4.5 使用柱状图显示数组数据 ……271
7.4.6 使用数组元素作为计数器 ……272
7.4.7 使用数组分析调查结果 ……273
7.5 引用和引用参数……276
7.6 把数组传递给方法……276
7.7 数组排序……279
7.8 数组查找：线性查找和折半查找……281
7.8.1 线性查找 ……281
7.8.2 采用折半查找对有序数组
进行查找 ……283
7.9 多维数组……288
7.10 （可选实例研究）关于对象的思考：
对象之间的协作 ……294
第8章 基于对象的编程 ……319
8.1 简介……319
8.2 用类实现抽象数据类型：时间……320
8.3 类作用域……327
8.4 成员访问控制……327
8.5 创建包……328
8.6 初始化类对象：构造函数……332
8.7 重载的构造函数 ……333
8.8 设置和读取方法……337
8.9 软件复用性……346
8.10 final实例变量 ……347
8.11 组合：对象作为其他类的实例变量 …348
8.12 包访问 ……351
8.13 this引用 ……353
8.14 终结函数 ……359
8.15 static类成员 ……360
8.16 数据抽象和封装 ……364
8.17 （可选实例研究）关于对象的思考：
开始对电梯模拟系统的类进行编程 …366
第9章 面向对象的编程 ……377
9.1 简介……378
9.2 超类和子类……379

9.3 protected成员 ……382
9.4 超类对象和子类对象之间的关系……382
9.5 子类中的构造函数和终结函数……388
9.6 子类对象向超类对象的隐式转换……392
9.7 利用继承性的软件工程……392
9.8 组合和继承的比较 ……393
9.9 实例研究：点、圆、圆柱体……393
9.10 多态性简介 ……400
9.11 类型域和switch语句……400
9.12 动态方法绑定 ……400
9.13 final方法和final类 ……401
9.14 抽象超类和具体类 ……401
9.15 多态性实例 ……402
9.16 实例研究：多态性在工资系统中的应用 ……404
9.17 新类和动态绑定 ……411
9.18 实例研究：继承接口和实现 ……411
9.19 实例研究：创建和使用接口 ……418
9.20 内部类定义 ……424
9.21 内部类定义的注释 ……434
9.22 基本类型的类型包装类 ……435
9.23 （可选实例研究）关于对象的思考：电梯模拟系统中继承性的应用 ……435
9.24 （选学）揭示设计模式：创建型设计模式、结构型设计模式及行为型设计模式简介 ……441
9.24.1 创建型设计模式……442
9.24.2 结构型设计模式……444
9.24.3 行为型设计模式……444
9.24.4 结束语……446
9.24.5 Internet和WWW资源……446
第10章 字符串和字符 ……455
10.1 简介 ……456
10.2 字符和字符串的基础知识 ……456
10.3 String类的构造函数……456
10.4 String类的length、charAt和getChars方法……458
10.5 字符串比较 ……460
10.6 String类的hashCode方法 ……464
10.7 在字符串中查找字符和子串 ……465
10.8 从字符串中抽取子串 ……467
10.9 字符串连接 ……468
10.10 String类的其他方法 ……469
10.11 String类的valueOf方法 ……471
10.12 String类的intern方法 ……473
10.13 StringBuffer类 ……475
10.14 StringBuffer类的构造函数 ……475
10.15 StringBuffer类的length、capacity、setLength和ensureCapacity方法……476
10.16 StringBuffer类的charAt、setCharAt、getChars和reverse方法……478
10.17 StringBuffer类的append方法 …479
10.18 StringBuffer类的插入和删除方法……481
10.19 Character类示例 ……482
10.20 StringTokenizer类 ……489
10.21 洗牌和发牌模拟……492
10.22 （可选实例研究）关于对象的思考：事件处理……496

第1章　计算机、Internet和Web

目标

- 理解计算机科学的基本概念。
- 了解不同类型的编程语言。
- 介绍Java的编程环境。
- 理解Java在开发Internet和Web分布式客户/服务器应用程序中扮演的角色。
- 用UML和设计模式介绍面向对象设计。
- 预览本书的其余各章。

我们的生命耗费在具体的细节上……简化，再简化。

——亨利·梭罗

高超的思想必须具有高超的语言。

——阿里斯托芬

语言的主要价值在于清晰。

——Galen

我的目标都是崇高的，我总有一天会实现。

——W. S. Gilbert

他在严密地组织思想，使之易于传播方面有出色的天分。

——Thomas Babington Macaulay

天哪！我认为翻译者是两者中最难被理解的人。

——Richard Brinsley Sheridan

提纲

1.1　简介
1.2　什么是计算机
1.3　计算机的组成
1.4　操作系统的演化
1.5　个人计算、分布式计算与客户／服务器计算
1.6　机器语言、汇编语言和高级语言
1.7　C++的历史
1.8　Java的历史
1.9　Java的类库
1.10　其他高级语言

1.11 结构化编程
1.12 Internet和WWW
1.13 典型Java环境的组成
1.14 关于Java和本书的通用注释
1.15 关于对象的思考：对象技术和统一建模语言UML简介
1.16 揭示设计模式：简介
1.17 本书概况
1.18 （选学）纵览利用UML进行面向对象设计的实例研究
1.19 （选学）浏览“揭示设计模式”的各个小节
小结 · 术语 · 自测题 · 自测题答案 · 练习

1.1 简介

欢迎使用Java!本书努力为用户创造一个集知识性、趣味性和挑战性为一体的学习体验。Java是功能强大的编程语言，它不仅可以使初学者感兴趣，同时也适用于有经验的程序员编制大型信息系统。本书不论对初学者还是有经验的程序员都是有效的学习工具。

一本书为何能使两种不同层次的人员都感兴趣呢？答案在于本书的核心侧重于运用结构化编程方法和面向对象方法中的成熟技术来获得简洁、清晰的程序。可以说，没有一个程序员从一开始就知道如何正确地编程。本书不但注重用清晰和直截了当的语句和丰富的示例来编写程序，而且提供大量Java程序及这些程序运行后的输出结果。我们通过完整的可运行的Java程序来教Java的特性。我们称为“生动活泼”的代码教学方法。这些例子可从以下三个地方获得——本书所附光盘、从www.deitel.com下载和交互式光盘产品：《Java 2 Multimedia Cyber Classroom》第4版。这张光盘包含本书近一半的练习答案，包括简洁的答案、小程序和许多完整的项目。如果你已购买了《The Complete Java 2 Training Course》第4版，因为它包含《Java 2 Multimedia Cyber Classroom》，那么你就有了这张光盘。

本书前面的章介绍计算机基础知识、计算机的编程方法以及Java编程语言。初学者可以通过学习这些章的内容为以后运用Java打下坚实的基础。有经验的程序员只需快速浏览这些章，并会发现以后章中Java的运用是严密且具有挑战性的。

许多有经验的程序员很欣赏本书对结构化编程方法的介绍。他们往往使用结构化编程语言(如C或Pascal语言)来编程，但因为从未正式学习过结构化编程方法，因此无法利用这些语言编写出最优的代码。在学习了第4章和第5章之后，他们的C或Pascal编程技巧必将有所提高。因此无论对于初学者或有经验的编程员，本书都是有益的、有趣的且具有挑战性的。

大多数人都了解计算机所能做的一些激动人心的事情。通过学习本书，读者将学会如何用计算机做事情，利用软件（即命令计算机执行操作或做出决定的指令集）控制计算机（通常指硬件）。Java是当今最流行的软件开发语言，可以免费从Sun公司的网站

java.sun.com/j2se

获得。本书是基于Java 2（标准版）平台，该平台描述Java语言、类库以及用于Java 2开发的工具。Sun公司提供了Java 2（标准版）软件开发工具箱（J2SDK），包括一套利用Java编写软件的必要工具。你可以从Sun公司的网站java.sun.com/j2se下载J2SDK的更新版。

计算机应用几乎在所有涉及的领域都在增加。在各种消费都高涨的年代，计算机的开销却由于硬件和软件技术的迅速发展而大幅度下降。20年前可能占据一大间房屋并花费几百万美元的计算机，如今可以存放在一个指甲大小的硅片上，并且只需花费很少的钱。有意思的是，硅是地球上最丰富的材料之一——它是普通沙子的一种成分。硅芯片技术使计算机变得非常便宜，以至于全球大约有几十亿台通用计算机运用在商业、工业、政府部门以及人们的日常生活中，而且近几年可能还会翻倍。

本书在以下几方面可能是具有挑战性的。读者可能花了几年时间学习第一门编程语言——C或Pascal语言，而且可能学习了结构化编程方法。本书不但介绍结构化编程方法，而且还介绍一个激动人心的新方法——面向对象编程方法。为什么本书要介绍这两种方法？这是因为我们相信面向对象的编程方法是未来的重要编程方法。在学习本书的过程中，读者将建立和使用许多对象，但对象的内部结构都采用结构化编程方法构建。同样，结构化的表达方式也最适于表达对象处理逻辑。

本书提供两种编程方法的另一个原因是，软件不断地从基于C语言的系统（主要是用结构化编程方法编写的）移植到基于C++和Java语言的系统中（主要运用面向对象的编程技巧）。因为C语言已经使用了30余年，因此存在许多“遗留下来的C语言代码”。读者学习了C++或Java语言后，将发现这些语言具有比C语言更强大的功能，就会选择C++或Java编写应用程序。他们往往先对已有系统进行转化，然后再使用C++或Java的面向对象编程方法，以充分利用这些语言的优点。通常，C++和Java之间的选择，往往取决于C++和Java哪一个更为简洁。

Java已经成为编写基于Internet和Intranet的应用程序，和其他一些用于网络通信设备的软件首选的编程语言。如果你的家庭中的立体音箱及其他家用设备利用Java与网络集成，你也不要感到奇怪。而且，你也不应对无线设备，如蜂窝电话、寻呼机和个人数字化助手（PDA），通过一种基于Java的网络协议（这些协议将在本书讲到）在无线Internet上通信而感到惊奇。

Java是一种最具吸引力的编程语言。2001年6月JavaOne™展示会宣布美国有56%的大学将Java作为一门必修的编程语言课程。并且，87%的美国大学开设了Jave课程。Java在中学也是有吸引力的。2003年，大学委员会将把Java作为高级计算机科学课程标准化。

Java在大规模应用领域的迅速发展，使得它不仅仅是只适用于制作“动态”网页的语言。Java成为许多组织编程的首选语言。

许多年以来，C和C++语言因其可移植性而倍受大学的欢迎。只要有C或C++的编译软件，这些语言软件的入门课程可以在任何硬件／操作系统上进行。但是编程世界变得日益复杂，且要求越来越高。如今，用户希望应用软件具有图形用户界面（GUI）；希望软件具有处理图形、图像、动画、音频及视频等多媒体的能力；希望应用软件可以在Internet上运行，并可与其他的应用程序进行通信；希望软件可以充分利用多线程的灵活性及性能优化（多线程让程序员指明应并行进行的多种动作）；希望应用程序能提供比C或C++更丰富的文件处理能力；希望应用程序不仅仅局限在自身的平台或局域网上，还可集成Internet上的组件和远程数据库；希望能利用预制软件组件快速、正确地编写应用软件；希望能够利用日益增长的可复用软件组件。程序员希望所有这些优点都可以方便地移植，以便在一个工作平台上编写的应用软件基本不做修改就可在不同的平台上运行（例如：运行在不同的操作系统、不同的计算机上）。Java满足了编程界的所有这些要求。

Java对大学课程具有吸引力的另一个原因是它完全面向对象。C++发展很快的原因是它将C语言编程扩展到面向对象的领域。这对于大量的C程序员来说是一个巨大的优势。C++既

包含ANSI/ISO C的功能，又包含面向对象的编程能力（ANSI是美国国家标准化组织，ISO是国际标准化组织）。在过去的几十年里，人们已用C语言开发了大量的代码。因为C++是C的超集，许多组织觉得它是理想的工具。程序员利用原有的C语言代码，基本上不做任何语言上的修改，即可在C++编译器中重新编译。程序员在学习掌握面向对象编程的同时，还可以继续编写与C类似的程序代码。只要时间允许，程序员可逐渐将遗留下来的C代码转化到C++，这样新的系统就完全是由面向对象的C++语言编写的了。一开始这一策略对很多组织都很有吸引力，但它有一个不利方面，那就是一旦采用此策略，这些公司将仍继续生产用类C代码的软件。这意味着他们并没有很快实现面向对象编程法的全部优点。许多公司希望立刻能100%地投入面向对象的开发，然而现实中堆积如山的遗留代码和采用C编程方法的诱惑却阻碍了这一进程。

Java是完全面向对象的编程语言，对软件工程技术提供强大的支持。在Java中很难编写类似C的所谓过程化程序，你必须创建和管理对象。错误处理建立在语言上。在C和C++编程中有许多复杂的细节妨碍程序员考虑全局结构，这在Java中是不存在的。这些特点对大学生来说很有吸引力，学生可以从一开始就学习面向对象的编程方法，并以面向对象的方式思考问题。

当然，这多少有些折衷。应用Java编写新应用软件的公司并不想将历史遗留下来的代码全部转成Java。因此，Java允许所谓的“本地码”，这意味着已有的C和C++代码可以集成到Java代码中。尽管显得很笨拙（并且确实如此），但它提供了一个可以解决大多数组织所面临问题的策略。

事实上，Java编译器可以从Sun公司的网站：http://java.sun.com/j2se免费获得，这对于面临预算困难和漫长预算计划周期的大学很有吸引力。并且由于bug修复和Java新版本的发展，这些可直接从Internet上获得，所以院校的Java软件能够一直保持最新版。

Java可以作为本书读者所学的第一门编程课程吗？作者认为完全可以。在写本书之前，Deitel & Associates公司的教师已向上百个班的几千名各种层次的专业人员，其中包括许多非程序员，讲授了Java语言，并发现非编程人员学习用Java编程比学习用C或C++编程更快。他们对使用Java提供的强大的图形、图形用户界面（GUI）、多媒体、动画、多线程及网络操作等功能抱有极大的兴趣，甚至可以在最初的几节课中就成功地构造大型的Java程序。

多年来，Pascal语言一直作为初级和中级编程课程中的首选语言。因为许多人认为对入门课程来说，C语言太难了。1992年我们出版了第1版的《C程序设计教程》，并希望以此取代大学课程中的Pascal语言。通过使用大学中沿用了多年的教学方法，但是融入C的概念而不是Pascal的概念，我们发现学生们在使用C语言时完全达到与Pascal语言同等的水平。但有一个明显的区别，学生们认为他们学习C语言对他们今后的工作是有价值的。工业界的客户很喜欢会用C语言的毕业生，因为这些人可以立即投入一些项目的工作而不必现花费大量的时间和金钱进行培训。

第1版《C程序设计教程》包含了60页介绍C++和面向对象编程方法的内容。当时考虑到虽然C++正日益流行，但在大学中用C++和OOP作为入门课程的编程语言仍为时尚早。

1993年，工业界对C++和OOP的兴趣突飞猛进，但我们仍然不认为大学会把C++和OOP作为入门课程的编程语言。因此，我们在1994年1月出版了第2版的《C程序设计教程》，并包含了300页有关C++和OOP的内容。1994年5月出版了第1版《C++程序设计教程》。这本950页的教科书更能适应许多在编程语言教学中处于领先地位的学校把C++和OOP作为大学入门课程

的要求。

1995年，作者仔细跟踪有关Java的介绍，并参加了1995年11月在波士顿举办的Internet会议。在Internet会议中心，Sun公司的代表准备在球形展厅介绍Java。随着演示的进行，人们发现Java在交互式、多媒体网页的设计中将扮演重要角色。随后又发现这种语言的需求潜力是巨大的，因为在这个充满图形、图像、动画、音频、视频、网络、多线程及协同计算的时代，Java是大学第一年编程语言入门课程的合适语言。当时，我们正在忙于编写第2版《C++程序设计教程》。在向Prentice Hall说明了Java将对大学课程产生巨大影响的观点后，大家一致同意延缓第2版《C++程序设计教程》的出版，以使第1版《Java程序设计教程》（基于Java 1.0.2）赶上1996年的秋季课程。

由于Java迅速演化到Java 1.1，在第1版发行不到一年，我们编写了《Java程序设计教程》第2版。世界上数百所大学和公司使用此版本。为了紧跟Java的发展，我们在1999年又出版了第3版。第3版主要是将内容升级到Java 2平台。

Java继续迅速发展，于是我们又编写了第4版——从第1版到第4版的出版，只经历了5年的时间。第4版是基于Java 2平台标准版（J2SE）。Java在最近的几年发展如此迅速，以致它现在已有另外两个不同的版本。Java 2平台企业版（J2EE）适用于大规模的分布式网络应用和他基于Web的应用。Java 2平台微型版（J2ME）是针对小设备（如蜂窝电话、寻呼机和个人数字化助手）的应用和其他内存受限制的应用而开发的。Java的内容如此丰富，所以一本书是不可能将它全部包含的。因此，我们在出版本书的第4版之时同时也出版了《Advanced Java 2 Platform How to Program》，这本书强调了使用J2EE来开发应用程序和提供了一些J2SE的高端内容。此外，这本书也包含了J2ME的丰富内容和无线应用的开发。

本书将使读者踏上一条充满挑战性和收获的学习过程。在学习过程中，读者可以将对Java和本书的想法通过我们的电子邮件地址deitel@deitel.com进行交流，我们也将尽快给予答复。

Prentice Hall的WWW站点www.Prenhall.com/deitel是我们的出版物网址，它包含教科书、专业书和交互式多媒体教室的光盘，完全训练教程（附有多媒体教程光盘和教材）、在线培训教程、电子文档、电子书刊以及所有这些产品的相关材料。该站点包含有关的常见问题（FAQ）、源代码的下载、勘误表、更新、补充的文本和例子、补充的自测题以及一些有关编程语言和面向对象编程技术的新发展。读者若想对本书作者及本公司有更多的了解，可参看本公司的站点www.deitel.com。

1.2 什么是计算机

*计算机*是快速的电子计算装置，它可以用比人快百万倍甚至数十亿倍的速度执行计算和进行逻辑判断。例如，当今许多个人计算机可以在一秒内执行上亿次，甚至几十亿次的加法。一个使用桌面计算器的人需用几十年时间才能完成一台强大的个人计算机在一秒内完成的计算。（考虑：你如何知道这个人的加法操作是否正确？你如何知道计算机的操作是否正确？）目前最快的*超级计算机*每秒可执行数千亿次的加法，这需要几十万人用一年的时间才能完成！而每秒可执行亿万次的计算机已在实验室中运行了。

计算机按照被称为*计算机程序*的指令集来处理数据。这些控制计算机正确执行有关操作的计算机程序是由*程序员*编制的。

组成计算机系统的各种各样的设备，如键盘、显示器、硬盘、存储器及中央处理单元等

叫做计算机硬件，在计算机中运行的程序叫做软件。随着个人计算机工业的产业化，近年来计算机硬件价格急剧下降。但是，软件开发技术却提高很少，软件的开发费用随着应用软件功能的增多和复杂度的上升逐步提高。从本书可以学习经过检验的软件开发方法，例如自顶向下逐步求精，模块化编程和面向对象编程方法，这些方法可以减少软件开发的费用。人们普遍认为面向对象的编程是个重大突破，它可以极大提高程序员的生产率。

1.3 计算机的组成

通常，不管外观如何不同，计算机系统由输入设备、输出设备、内存储器、运算器、中央处理单元和外存储器六大逻辑部件组成 。如下所示:

1）输入设备

这是计算机的输入部件。它通过输入设备获取信息（数据或程序）并将这些信息存放在处理单元以便处理。常用的输入设备有键盘、鼠标和磁盘等。将来还可以通过语音、电子扫描图像和视频录像等方式输入信息。

2）输出设备

这是计算机的输出部件。它将计算机运算处理的结果以人们需要的形式输出到输出设备上。常用的输出设备有显示器、打印机、音箱、磁盘或磁带等。也可以将输出信息转换为控制信号，用来控制其他设备。

3）内存储器

这是计算机快速存取信息的存储功能部件。用于保存通过输入设备输入的信息，以便需要时可立即提供给计算机进行处理。同时，也用于存放处理过程中产生的中间结果和最终的结果，直到通过输出设备进行输出处理。计算机内存储器又叫内存、主存或随机存储器(RAM)。

4）运算器

它是计算机的生产部件，又叫算术逻辑运算单元（ALU)。是计算机执行加、减、乘、除、比较和逻辑与、或、非等算术逻辑运算的功能部件。它包含判别机制，允许计算机比较内存中的两个项是否相等。

5）中央处理单元（CPU)

它是计算机的控制部件，由它产生控制命令，控制及协调计算机各部件步调一致地自动工作。例如由CPU控制输入设备何时应将数据存入内存储单元，控制ALU何时从内存单元中读出数据进行计算，控制输出部件何时从存储单元中读出数据送到输出设备。

6）外存储器

这是计算机的外部存储器，例如硬盘、磁带等存储器。其特点是价格便宜、容量大、可永久保存信息。可将暂时不用的程序或数据放在计算机外存储器中长达数小时甚至数年。存取外存储器中的信息比存取内存单元中的信息速度慢。但其价格比内存要低得多。计算机的外存储器又叫辅助存储器或二级存储器。

1.4 操作系统的演化

早期的计算机一次只能完成一项工作或任务，称为单用户批处理。它每运行一个程序会处理成组或成批的数据。在早期的系统中，用户常常将他们的工作以穿孔卡片的形式提交给计算机中心。对输出的结果他们常常不得不等几个小时甚至几天。

人们开发了称为操作系统（operating system）的软件系统，使用户能够更方便地使用计算机。早期的操作系统以顺序的方式进行任务管理。这样减少了操作系统在各个任务间来回切换的时间，因此提高了计算机的处理能力。

随着计算机软件资源的丰富和硬件配置的迅速增强，单用户批处理操作系统在有效地利用计算机资源方面存在明显的不足。于是人们研制了多个进程或多任务的多道程序操作系统，该类操作系统可对多任务实现同时操作——通过在竞争的任务间共享计算机资源。在早期的多道程序操作系统中，用户仍然将他们的工作以穿孔卡片的形式提交给计算机，而且也需要为结果等待几个小时甚至几天。

20世纪60年代研究并发布了分时操作系统。分时操作系统是多道程序操作系统的一种特殊类型。在分时操作系统的管理下，用户通过终端（通常是带键盘和显示屏的设备）与计算机进行交互。在典型的分时计算机系统中，同一时间可以有几十甚至几百人同时通过终端使用计算机。尽管计算机在某一时刻并不同时运行所有的用户程序，而只运行一个用户程序的一部分，经过一个时间片后又转向下一个用户程序。但因为计算机速度非常快，可能每秒为每个用户提供几次服务，能够保证有足够快的响应时间，所以在用户看来，就像是同时运行他们各自的程序似的。分时方式的一个优点是用户几乎可以立刻得到对计算机请求的响应，而不需要等待很长时间。同时，如果一个特定用户当前空闲，计算机可以继续为其他用户服务而不是等待这个用户。

1.5 个人计算、分布式计算与客户／服务器计算

1977年，Apple公司使个人计算机流行起来。最初，这只是电脑迷的梦想，即人们可以购买足够便宜的计算机用于个人生活或商业用途。1981年，世界上最大的计算机生产商IBM推出了IBM个人计算机。几乎一夜之间，他个人计算机用于商业、工业和政府管理的想法成为现实。

但是，这些计算机都是各自“孤立”的个人计算机，用户在自己的机器上完成工作后再用磁盘分享信息。尽管早期的个人计算机还未强大到可使几个用户分时共享，但它们仍可以通过计算机网络连在一起，例如通过电话线，或通过公司内部的局域网（LAN）形成分布式计算模式。这样，公司的计算就无需在计算机中心进行操作，而是通过网络传送到公司工作实际执行的地点。个人计算机不但完全能够处理个人用户的要求，而且可以处理通过网络传送的信息，并完成基本通信任务。

如今，性能强大的个人计算机可以与十年前几百万美元的计算机相比。功能最强大的桌上型电脑——工作站为个人用户提供了许多功能，信息可以很方便地通过计算机网络实现共享。作为文件服务器的计算机，可以给分布在网络上不同站点的客户计算机提供共享的公用程序和数据，这就是客户／服务器计算模式。C和C++已成为而且现在仍然是常用的操作系统的编程语言。它们也还是计算机网络软件、分布式客户／服务器应用程序以及Internet和Web应用程序的编程语言，虽然Java现在正成为这些领域的主要编程语言。许多编程人员已经发现，用Java编写程序比用C或C++更有效率。当前流行的操作系统，像UNIX、Linux、MacOS、Windows和Windows 2000都提供了本节所讨论的各种功能。

1.6 机器语言、汇编语言和高级语言

程序员用各种编程语言书写指令，有些可以直接被计算机所理解，而有些则需要中间的

解释或翻译步骤。目前正在使用的计算机语言有几百种，可将它们分成三大类：

1）机器语言

2）汇编语言

3）高级语言

任何计算机都能直接识别仅用它自己的机器语言编写的指令。机器语言是计算机的“自然语言”。它是由计算机的硬件设计定义的，通常由数字串（最终缩减为0和1）组成，用来指示计算机执行基本的操作。机器语言是与机器相关的，即某种机器语言仅可用在某特定型号的计算机上。用机器语言编写程序既繁琐费时、容易出错，又难以记忆、识别，人们用起来十分困难。以下为一段机器语言程序，该程序将每天的加班费和基本费用相加并存储总的费用。

```
+1300042774
+1400593419
+1200274027
```

这种语言书写的程序是很难理解的。随着计算机的普及，很明显程序员用机器语言编程既慢又繁琐。人们研制利用类似英语缩写的字符串来代表机器语言指令进行程序设计的语言，称为汇编语言。汇编语言程序必须通过汇编器的翻译程序转换为机器语言表示的目标程序，才能在计算机中执行。下面是用一种汇编语言书写的程序片段，同样完成将每天的加班费和基本费用相加，并存储总的费用的功能，但这比上面列举的用机器语言书写的程序段要清楚得多且容易理解。

```
LOAD        BASEPAY
ADD         OVERPAY
STORE       GROSSPAY
```

尽管这种代码容易被人们理解，但是只有被翻译成机器语言后才能被计算机识别。

汇编语言的出现使计算机应用得到进一步的发展，但汇编语言仍然是面向机器的语言，即使一个最简单的任务也需要多条指令才能完成。一般用户掌握和使用这种语言仍比较困难。为了方便编程和加快计算机的应用，人们研制出高级语言。高级语言使用一条简单的语句就可以完成由许多条汇编语句才能完成的任务。将高级语言转化为机器语言是通过执行称为编译器的翻译程序实现的。高级语言采用类似日常英语及数学符号来书写语句，组成程序。用高级语言实现计费功能，只需用如下一条语句即可完成。

```
grossPay = basePay + overTimePay
```

显然，在程序员看来，高级语言比机器语言和汇编语言都更有吸引力，易学、易用、易于理解。C、C++和Java都是当代功能强、应用得最广泛的高级语言。

要使高级语言源程序在计算机中运行，必须先由编译器翻译成机器语言目标程序。若使用解释程序，则可直接在相应解释程序支持下运行解释型的高级语言源程序而不再需要将其编译成机器语言目标程序。尽管编译后的可执行程序比在解释程序支持下运行高级语言源程序要快，但解释程序在程序开发环境中仍然很流行。在这种环境下，程序要被频繁地重新编译，因为要加入新的功能和修改错误。一旦程序开发完成，编译版本运行起来更有效。在学习Java的时候，读者会看到，解释程序在帮助Java实现其在各种不同的平台上的可移植性目标方面扮演着十分重要的角色。

1.7 C++的历史

C++是由C发展而来的，而C则是在以前的BCPL和B两种语言的基础上发展起来的。

BCPL是由Martin Richards在1967年开发，用来编写操作系统软件和编译器的语言。Ken Thompson借鉴了BCPL的许多特点开发了B语言，并于1970年在贝尔实验室的一台DEC PDP-7上实现了最早版本的UNIX操作系统。无论是BCPL还是B语言都是无类型的语言，即每个数据占用存储单元中的一个“字”，例如，必须由程序员来决定将数据当成实数还是整数。

C语言是1972年由贝尔实验室的Dennis Ritchie在B语言的基础上开发的，并在一台DEC PDP-11计算机上实现。C语言吸收了BCPL和B语言的许多重要概念，同时增加了数据类型和其他特征。C语言最早是作为UNIX操作系统的开发语言而被广泛了解的。如今，几乎所有新的大型操作系统都是由C或C++编写的。在过去的20多年里，大多数计算机都可以运行C程序。C程序不依赖于计算机。因此，只要仔细地设计就可以使C程序方便地移植到大多数计算机上。

20世纪70年代后期，C已发展为现在称为“标准C”或“Kernighan and Ritchie C”。1978年由Prentice Hall出版的《C程序设计语言》，引起了对这种语言的广泛注意，并成为畅销的计算机科学书籍。

在不同计算机硬件平台上广泛地使用C语言导致了很多不同的版本。这些版本虽然相似，但却不兼容。这对于需要编写在不同平台上运行的可移植程序的程序员来说是很麻烦的问题，他们需要一个C的标准。1983年，美国计算机和信息处理（X3）标准化委员会成立了X3J11技术委员会，以便“为这种语言提供一种没有歧义和不依赖于机器的定义”。1989年，此标准获得通过。ANSI和国际标准化组织（ISO）合作，促进世界范围内C的标准化，并于1990年出版了最终的C标准ANSI / ISO 9899:1990。1988年出版了Kernighan和Ritchie的第2版《C程序设计语言》，反映了目前被广泛使用的C语言版本ANSI C [Ke88]。

由C发展和扩充而来的C++，是Bjarne Stroustrup于20世纪80年代早期在贝尔实验室开发出来的。C++提供了一些使C更加完备的特性，但最重要的是它提供了面向对象编程的功能。C++也被ANSI和ISO委员会进行了标准化。

这是软件世界的一场革命。在市场对新的、功能强大的软件的需求急剧上升的同时，快速、正确、经济地编写软件仍是一个难以达到的目标。对象主要是一些可复用软件的组件，它们是真实世界事物的模型。软件开发者发现，使用模块化、面向对象的设计和实现，可以使软件开发小组的效率比使用以前流行的技巧，例如结构化编程方法，要提高很多倍。面向对象的程序更容易理解、修正和改变。

已有的面向对象的语言还有施乐公司的Palo Alto实验中心（PARC）开发的Smalltalk。Smalltalk是一种纯粹的面向对象的语言，一切事务都严格地是对象。C++是一种混合语言，它可以用类C风格或面向对象的风格或两者组合的形式编程。

1.8 Java的历史

微处理器革命最重要的贡献也许就是它使全世界拥有几亿台个人计算机成为可能。个人计算机对个人以及对企业的生产管理方式都产生了深刻的影响。

微处理器产生深刻影响的另一个领域是智能型电子装置。基于这一点，Sun公司于1991年在公司内部投资了一个名为Green的研究项目。该项目组开发了一种以C和C++语言为基础的语言。它的创造者James Gosling根据他在Sun公司的办公室外的一棵橡树（oak），而称其为Oak语言，后来发现已有一种称为Oak的计算机语言。当一些Sun公司的员工到当地一家咖啡店时，有人提议将该语言命名为Java，从而使Java这个名字一直延续至今。

但Green项目遇到了一些困难，市场对智能型电子装置需求的上升率并不像Sun公司所期

盼的那样快，更糟的是Sun公司参加竞争的一个重要的销售合同被另一个公司得去了。此时，Green项目几乎处于被取消的境地。但很幸运的是，1993年万维网疯狂地流行起来，Sun公司发现了用Java创建具有动态功能的网页的潜在需求。这给该项目重新注入了生机。

Sun公司于1995年5月在一个重要的会议上正式发布了Java。这样的事通常不会引起广泛的注意，但是由于万维网的商业利益，Java立即引起了商业界的极大兴趣。目前Java被广泛应用于创建具有动态的、交互内容的网页，开发大规模企业应用程序，增强万维网服务（提供我们在Web浏览器所看到的内容）的功能，向消费者设备（例如蜂窝电话、寻呼机、个人数字化助手）和许多其他应用提供应用程序。

1.9 Java的类库

Java程序由类组成，类则由方法构成。方法用来执行任务，并在完成任务时返回信息。用户可以编写类和方法，并由此建立一个Java程序。但在编程过程中，他们还应尽量充分利用Java类库中已存在的丰富的类和方法。类库就是Java API(Application Programming Interface，应用程序接口)。因此学习Java语言实际上包括两个方面：一方面是学习用Java语言编写自己所需的类和方法，另一方面是学习如何利用Java类库中的类和方法。本书将讨论许多类和方法。类库主要是由一些编译器生产商提供的，但也有许多类由独立的软件商（ISV）提供。而且许多类库可以作为免费软件和共享软件从Internet和WWW下载。读者可以下载和使用免费的软件，不受任何版权限制。读者也可以下载免费的共享软件并可使用该软件，但仅限于个人免费使用。如果读者想将共享软件定期使用或用于商业目的，必须向软件的版权所有者交纳一定的费用。

许多免费软件和共享软件的源代码都是公开的。这些公开的源代码都可以从Internet上免费获得，读者可研究学习这些代码，验证是否实现了它们的功能，甚至可以修改这些代码。公开源代码的产品有可能通过用户的改进方案而进一步发展。Linux操作系统就是一个公开源代码的软件产品。

软件工程评述1.1 使用组件方法创建程序，可以避免从头做所有的编程工作。使用已存在的组件——称为软件复用，这是面向对象编程的中心环节。

[**注意**：贯穿全书会有许多软件工程评述，解释那些对软件系统，尤其是大型的软件系统的总体结构和质量有所影响的概念。我们还将突出良好的编程习惯（有助于编写结构更清楚、更容易理解、更容易调试和维护的程序）、常见编程错误（帮助避免同一错误）、提高性能的建议（帮助你编写运行得更快、占用内存更少的程序）、可移植性提示（帮助编写无须修改或只做微小修改就可在不同计算机上运行的程序）、测试和调试提示（帮助消除程序中的错误，更重要的是有助于读者从一开始就编写正确的程序）和外观和风格评述（帮助设计图形用户界面的“外观”与“感觉”，更容易使用）。这些技术和习惯只是指导，毫无疑问，读者会有自己偏爱的编程风格。]

软件工程评述1.2 用Java编程，通常可以使用以下组件：类库中的类和方法，自己创建的类和方法以及他人创建而可由你使用的类和方法。

创建用户自己的类和方法的好处是可以正确了解它们是如何工作的，并可以检查Java代码；不利的方面是设计和开发新的类和方法要花费大量的时间和精力。

提高性能的建议1.1 使用Java API类库和方法而不是自己编写，可以提高程序运行的性能，因为类库中的这些类和方法都是经过精心地和仔细地设计，其运行效率高、质量高。这种方法也有助于你加速开发程序原型的速度（即，开发一个新的并且可以运行的第一版本程序所花的时间）。

可移植性提示1.1 使用Java API类库和方法而不是自己编写，可以提高程序的可移植性，因为这些类和方法包含在几乎所有的Java软件中（如果是同一版本）。

软件工程评述1.3 扩展类库的可复用组件一般可通过Internet和万维网得到。大多数类库都提供源代码并且是免费的。

1.10 其他高级语言

目前人们开发了几百种高级语言，但只有一小部分被广泛应用。Fortran（FORmula TRANslator)语言是由IBM公司于1954～1957年间开发并推出的，用于需要进行复杂数学计算的科学和工程领域。Fortran语言至今仍被广泛应用。

COBOL（COmmon Business Oriented Language）语言是1959年由一群计算机生产商、政府及工商业界的用户联合开发的。COBOL主要用于需要精确及有效处理大量数据的商业领域。如今，仍有大约一半的商业软件是用COBOL编写的，并且约有100万人正在从事编写COBOL程序的工作。

Pascal语言和C语言几乎同时问世。Pascal是由Nicklaus Wirth教授设计用于学术研究的语言。在下一小节将进一步讨论Pascal。

Basic语言于1965年在Dartmouth 大学开发，它是做为一门帮助新手熟悉编程的简单语言。比尔·盖茨在多种早期的个人计算机上实现了Basic。今天，微软公司——比尔·盖茨创建的公司成为软件开发组织的领路人。

1.11 结构化编程

20世纪60年代，许多大型软件的开发工作都遇到了严重的困难，软件开发的进度安排往往被拖延，花费大大超过预算，且最终产品并不可靠。人们终于认识到软件开发比他们设想的要复杂得多。20世纪60年代的研究活动导致了结构化编程方法的发展。按照结构化规则编写程序，比用非结构化编程方法编写的程序结构更清楚、更容易测试和调试、更易于修改。第4章和第5章将讨论结构化编程方法的一些原则。

这种研究的一个成果是Nicklaus Wirth于1971年开发出以17世纪数学家和哲学家Blaise Pascal命名的Pascal语言。设计Pascal语言的目的是用于大学中教授结构化编程方法，并很快成为大学里首选的编程语言。但这种语言缺少许多特性，使其不能用于商业、工业以及政府部门的应用中，因此它在这些领域并没有被广泛接受。

Ada编程语言是在美国国防部（DOD）的资助下于20世纪70～80年代早期开发出来的。DOD使用了上百种独立的语言开发其庞大的指挥控制软件系统。DOD需要一种可以满足大多数需求的语言。他们选择Pascal语言作为基础，但最终的Ada语言与Pascal之间有很大的差别。Ada语言是以诗人Lord Byron的女儿Ada Lovelace女士命名的。Lovelace女士在19世纪早期编写了世界上第一个计算机程序而获得广泛的尊敬。这个程序是为Charles Babbage设计的机械

计算装置“解析引擎”编制的。Ada语言的一个重要的功能是支持*多任务*，允许程序员指定许多并行发生的动作。大多数广泛使用的高级语言(包括C和C++)只允许程序员编写在一个时刻执行一个任务的程序。而Java却不同，通过使用称为*多线程*的技术，程序员可以编写并行行为的程序。[注意：大多数操作系统提供针对单独平台的库（有时称为*依赖于平台的库*），这种库使C和C++这样的高级语言可以在程序中指定并行执行的许多操作。]

1.12 Internet和WWW

Internet是30年前由美国国防部提供资金开发的。最初设计只是用来连接大约十几所大学和研究机构的主要计算机系统。今天，连入Internet的计算机数量已达几亿台。

随着WWW的应用——它允许计算机用户查找和查看所有主题的基于多媒体的文档，Internet的使用明显地呈爆炸式增长，看来它将成为世界上最主要的通信机制。

Internet和WWW肯定会被列入人类最重大及最深远的创造之列。过去，大多数计算机应用程序运行在没有与其他机器连接的单机上。今天的应用程序可以实现在全球数亿台计算机之间的通信。Internet集成了计算与通信技术。它使得工作更容易，全世界的信息可以及时、方便地获得。它使得个人和小型本地企业走向世界，而且正在改变商业运作的方法。人们能够找到几乎任何产品或服务的最优价格。特殊兴趣的社团可以通过Internet保持联系。科研工作者可以通过Internet了解最新的进展。

《Java程序设计教程》（第4版）介绍了一些编程技术，使得Java程序利用Internet和WWW与其他应用程序进行交互。这些功能，以及在我们配套出版的《Advanced Java 2 Platform How to Program》中所讨论的功能，使得Java程序设计人员可以开发用于工业领域的企业级分布式应用程序。用Java编写的程序可以在任何计算机平台上运行，大量节约系统开发时间和公司的费用。如果你最近一直听到许多有关Internet和WWW的事情，如果你对开发在Internet和WWW运行的程序有兴趣，学习Java将会为你提供极具挑战性和高回报的职业机会。

1.13 典型Java环境的组成

Java系统一般是由环境、语言、Java应用程序接口（API）以及各种类库构成。以下讨论将解释如图1-1所示的典型的Java程序开发环境。

Java程序的运行一般需经过五个步骤：*编辑*、*编译*、*装载*、*校验*和*运行*，如图1-1所示。我们所讨论的这些概念都是基于本书所附配套光盘中的Java 2*软件开发包*（J2SDK）而言的。将光盘中的J2SDK正确地安装后就可以正确地编译和执行Java程序。[注意：尽管你的系统可能与图1-1所示的环境很相似，但是对于不使用UNIX/Linux、Windows 95/98/ME或Windows NT/2000的读者最好是参考系统手册关于Java环境的部分或请教教师在该环境中如何实现上述任务。]

步骤1编辑一个文件，可以通过一个*编辑程序*完成（通常称为*编辑器*）。程序员用编辑软件编写一个Java源程序并做必要的修改。然后该程序以文件形式存储在计算机的外存储器(如硬盘)中。Java源程序文件的扩展名为`.java`。在UNIX/Linux系统中广泛使用的编辑软件是`vi`和`emacs`，在Windows 95/98/ME和Windows NT/2000上提供有简单编辑软件，如DOS的Edit编辑命令，Windows的Notepad等。在Java集成开发环境的编程环境（IDE）中，例如Forté for Java共享版、NetBeans、Borland的JBuilder、Symantec的Visual Cafe和IBM的VisualAge都集成了一个内嵌的编辑器。在本步骤中假设读者已经知道如何去编辑一个程序。

[**注意**：Forté for Java共享版是用Java编写的，对于非商业使用是免费的。它包括在随书的光盘中。Sun公司大约每年更新两次，最新的版本可以从

www.sun.com/ forte/ffj

下载得到。Forté for Java共享版可在大多数主要的平台上执行。本书是基于Java 2开发环境编写的，与Forté for Java共享版是独立不相关的。书中程序可在大多数Java集成开发环境中正确运行。]

步骤2（在第2章和第3章中继续讨论）程序员通过执行命令javac来编译程序。执行命令，Java编译器将Java源程序翻译成字节码——能被Java解释器理解的一种语言。如要编译名为Welcome.java的Java程序，可在系统命令窗口（例如Windows下的MS-DOS提示、Windows NT/2000下的命令行提示符或UNIX/Linux的shell的提示）键入如下命令：

```
javac Welcome.java
```

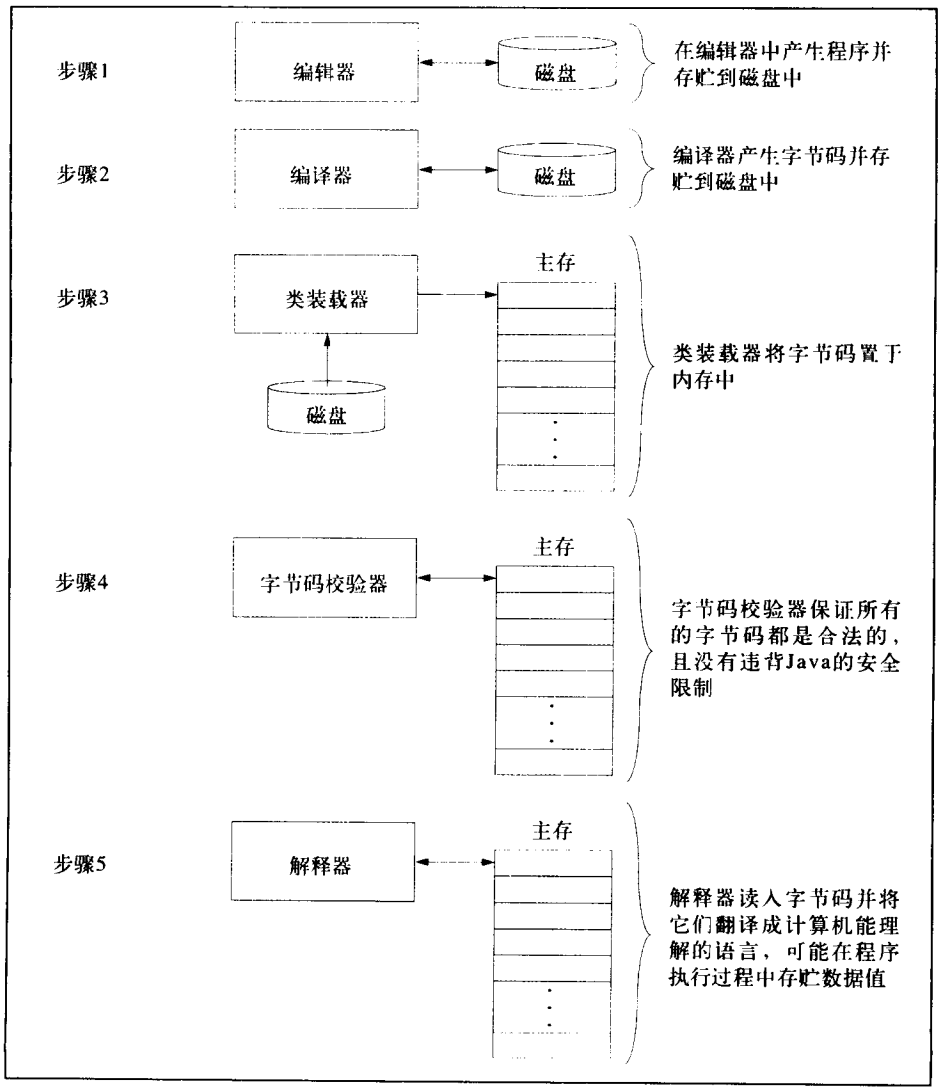

图1-1 典型的Java环境

如果程序编译通过，编译器将产生一个称为Welcome.class的文件，它包括了在执行过程中能被解释的字节码。

步骤3 *装载程序*。程序执行前必须由*类装载器*将其从外存装载到内存，即将包含字节码的.class文件调入内存。.class文件可以从系统的硬盘或通过网络（例如校园网或公司的局域网，或Internet）装载。有两种类型程序的.class文件由类装载器装载：*应用程序*及applet。应用程序，例如字处理程序、电子表格程序、画图程序、e-mail程序等通常存储和运行在本地用户机器上。applet是小程序，通常存储在用户经过WWW浏览器连接的远程计算机上。applet被下载到浏览器中，在浏览器里解释执行，并当运行结束后丢弃。当再次执行同一applet时，用户必须用浏览器访问同一地址，重新将此程序载入浏览器。

应用程序可以载入内存，通过命令java使用Java*解释器*执行。用以下命令执行Welcome程序：

```
java Welcome
```

这将引用Java解释器，并促使装载器装载Welcome程序中所需的信息。［**注意**：许多Java程序员把解释器称为Java*虚拟机*或JVM（Java virtual Machine, JVM）。］

将Java applet载入浏览器时，例如Netscape的Navigator，微软的Internet Explorer，类装载器也将被运行。浏览器用来浏览WWW上的HTML（超文本标记语言）文档。HTML将文档格式化使其能被浏览器显示（我们将在第3.4节介绍HTML和其他Internet编程技术。请参阅《Internet和WWW程序设计》(第2版)一书）。HTML文档可以含有Java的applet。当浏览器载入含有applet的HTML文档时，便运行Java类装载器载入applet（通常是从存放HTML文档处）。支持Java的浏览器有内置的Java解释器。一旦applet被装入，浏览器的Java解释器立刻执行此applet。applet还可通过使用Java开发工具（J2SDK）中的appletviewer工具软件来运行。J2SDK是一个工具集，它包括Java编译器（javac），解释器（java），appletviewer以及其他一些Java程序员要使用的工具。像Netscape Navigator和微软的IE一样，appletviewer也要由HTML文件来调用applet。例如，Welcome applet的HTML文件是Welcome.html，那么可用如下appletviewer命令：

```
appletviewer Welcome.html
```

在Welcome applet中所使用的信息将由该命令调用类装载器装载。appletviewer可认为是一个最小的浏览器，它只知道如何解释applet而忽略文档中的其他的HTML。

在字节码被Java解释器或appletviewer执行之前，它在步骤4要先通过字节码*校验器*的检查，以确保字节码有效且不违反Java的安全规则。Java必须实施安全保护，因为通过网络得到的Java程序可能会对用户的文件或系统造成破坏，如计算机病毒。也要注意从网上下载的应用程序中的类也需要进行代码验证。

最后，在步骤5中，计算机在CPU的控制下，一次解释执行一个字节码，从而执行程序所指定的动作。

程序很少在第一次调试时就通过，因为本书讨论的各种类型的错误都可能导致上述某一步的失败。例如，程序中可能出现除零错误（和在数学中一样，这在Java中也是非法操作）。这时计算机将给出一条错误信息。程序员将回到编辑步骤，对程序做必要的修改，然后再重新完成剩下的步骤以确保修改正确。