信息管理与计算机应用专业实用教程系列

Java程序设计

苏俊◎编著

机械工业出版社
China Machine Press

无线局域网安全——方法与技术

马建峰　朱建明　赖晓龙　牛广平　等编著

机 械 工 业 出 版 社

本书是在对“863”高科技项目——宽带无线 IP 网络系统安全技术(编号:2002AA143021)研究的基础上编写的。全书在介绍有关信息安全基本理论和方法的基础上，重点研究了无线局域网中的有关安全技术。不仅介绍了基本的方法与技术，还介绍了国内外最新的研究成果和发展动态，以及作者在研究过程中提出的一些安全方案和技术。全书分两篇，共 12 章，前 5 章着重介绍网络安全的基本理论，第 6 章到第 12 章介绍无线局域网中的安全方法与技术。附录 A 介绍了可证安全协议分析与设计的思想，附录 B 介绍了 WAPI 认证机制的性能和安全性分析，附录 C 介绍了一种新的增强的无线认证基础设施 EWAP。

本书可作为计算机、信息安全、信息对抗等专业高年级本科生或研究生的教学用书，也可作为相关领域的研究和工程技术人员的参考用书。

图书在版编目（CIP）数据

无线局域网安全：方法与技术/马建峰等编著 .—北京：机械工业出版社，2005.8
ISBN 7-111-16565-9

Ⅰ.无 … Ⅱ.马 … Ⅲ.无线电通信—局部网络—安全技术—高等学校—教材 Ⅳ.TN925

中国版本图书馆 CIP 数据核字（2005）第 049084 号

机械工业出版社(北京市百万庄大街 22 号 邮政编码 100037)
策 划：胡毓坚 责任编辑：陈振虹 版式设计：霍永明
责任校对：李汝庚 责任印制：陶 湛

北京铭成印刷有限公司印刷

2005 年 8 月第 1 版第 1 次印刷
787mm × 1092mm 1/16 · 20 印张 · 488 千字
0001—5000 册
定价：28.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
本社购书热线电话(010)68326294
封面无防伪标均为盗版

前 言

在计算机技术和管理方法日益普及和实用化的今天，无论哪一门学科的教学、研究和交流等工作都离不开计算机应用。计算机技术从枯燥乏味的二进制机器操作逐步发展到具有继承、封装等特点，进而提供了更适合人类思维方式的面向对象技术。Java技术以其独特的技术优势向人们展现了一种表达思想的方式，这就是尽可能地按照人们的思维习惯来更好地组织和管理资源。所以说，Java不仅仅是一种计算机的编程语言，也是一个规范管理资源的平台。

随着需要解决的问题日益复杂和庞大，Java从技术层面和管理层面提供了更方便、灵活和规范的表述能力。正因为如此，围绕Java技术，我们在信息管理、计算机应用等相关专业开设了一系列课程，并且取得良好效果。为了适应新形势的要求，在机械工业出版社华章分社的策划和大力帮助下，针对Java平台编写了以下4本教材。

(1)《Java程序设计》

这本书主要介绍Java程序设计方法，深入浅出、示例性地介绍Java常用编程技术，目的是使读者能够使用Java语言熟练设计和编写实用程序。

(2)《Java与网站编程技术》

这本书主要介绍Servlet、JavaBean和JSP等建设网站的知识和技术，目的是使读者能从信息处理的角度搭建Internet/Interanet应用。

(3)《Java与数据库技术》

数据库技术始终处于信息处理的中心，在构建企业应用时如何有效地综合使用Java与数据库技术是非常重要的话题。本书的目的是使读者能融合Java和数据库系统的优势，构建更实用的企业应用。

(4)《Java与系统设计》

建设企业应用是一项复杂而规范的软件开发任务，从需求分析、构建模型到具体实施是一条分工明确、操作性强的软件生成流水线，本书循序渐进地、示例性地讲解UML的知识和技术，并由UML生成有效的Java代码，目的是使得读者了解从系统设计到应用实施的过程。

这些教材原理和操作技巧并重，并选用目前流行版本的软件作为实验平台它们既可以作为大学教材，也可以作为相关研究人员更新知识的参考书。这些教材的每一本书既可以

单独使用，也可以整体使用。我们衷心希望它们真正成为读者学习知识的良师益友、为我国信息化教育和普及做出贡献！

本书特点与内容

学习Java程序设计一定要用面向对象的思维去学习，原因很简单，因为Java是新一代面向对象的程序设计语言。

当人类试制超音速飞机时，在模拟实验中，飞机在速度接近音速时常常被莫名其妙地撞得粉碎。后来人们发现这是一个看不见的“音障”在起作用，如何打破这个音障成为研制超音速飞机的关键。现在看来，解决问题的方法很简单，在飞机前部安装一个尖尖的金属棒，由它首先在音障上捅个小窟窿，那么整个飞机就可以突破音障的限制。

在学习Java程序设计时，也要借鉴这个方法。尤其对于初学者来讲，要以示例（金属棒）来学习Java程序设计。

首先Java语言规范提供了非常多的内容，我们不可能在一本教材中介绍所有内容，字典式的学习方式也会使得学习效果不明显，所以本书主要介绍Java的常用类，并通过示例把它们的作用展示出来。只要读者掌握了本书所介绍的内容，就具备了良好的Java编程能力；同时我们也掌握了一种学习方法，当要应用那些本书没有介绍到的Java类时，也只是存在量的问题了。

人们在认识一个对象时，总是考虑其静态信息、动态操作以及约束条件。同样在本书中，我们也希望大家习惯于从继承关系、成员属性、构造方法和成员方法4个方面来学习每一个Java对象和类。

伟大的科学家牛顿曾经说过：我之所以看得远，是因为我站在巨人的肩上！Java基础类封装了大量的实用程序类，本书在充分介绍了基本程序设计方法之后，也更多地介绍这些类的使用。让我们站在Java基础类的肩上，设计出更多解决新问题的实用程序。

全书分为9章。第1章Java绪论；第2章Java与面向对象技术；第3章Java语言的编程基础；第4章Java的常用基础类；第5章Java数据库编程技术；第6章GUI设计；第7章Java多媒体技术；第8章Java多线程技术；第9章Java与Socket编程。这些内容以JDK1.5.0为上机实验操作平台。本书以一个学生信息管理系统为项目设计实例，根据每章所介绍的内容逐步丰富和完善这个实例。

这里要特别感谢机械工业出版社华章分社温莉芳总编以及有关同志，她们在本书的策划、编写过程中给予了大力的支持。另外感谢刘富强、杜宏路对于本书形成所做的工作。在本教材编写的过程中得到了许多老师和同学的支持和配合，在此一并致谢！

由于作者水平有限，书中错误和不当之处在所难免，欢迎读者和广大同仁批评指正。

编　者

2005年3月于北京海淀

目 录

第1章

Java绪论

学习指南

面对各种亟待解决的问题，人们都要有解决的步骤和方法。如何把这些步骤和方法准确无误地告诉给计算机系统，并且让计算机系统帮助人们来解决问题，这是程序设计的任务。从实现环节上讲，程序设计包括算法设计和数据存储，也就是操作与数据之间的关系，这里倡导“程序＝数据结构＋算法”的理念。

人与计算机之间的交互存在各种各样的语言。计算机程序设计语言经历了从侧重计算机理解到逐步过渡到面向人们理解的发展过程。Java语言是一种成熟的、面向对象的、更适合人们思考习惯的高级计算机程序设计语言。计算机程序设计是信息处理的基础，所以认真、高效地学习和掌握计算机程序设计知识是非常必要的。

本章从介绍计算机程序设计的基本概念开始，逐步介绍Java语言的基本情况、开发环境、运行环境以及Java程序的组织方式。

本章学习的难点在于理解下列问题:

(1) 计算机程序的基本组成。

(2) 了解设计数据结构和算法的重要性。

(3) Java语言的特点。

(4) 支持Java程序运行的外部环境。

1.1 计算机程序设计概述

目前的计算机体系结构都是采用冯·诺伊曼体系结构，它包括两部分内容:

(1) 一台计算机由5个基本部件组成: 输入设备、输出设备、中央控制器、运算器和存储器。中央控制器（Central Processing Unit，简称为CPU）是计算机系统的核心，它会根据计算机所要处理的任务合理和高效地指挥其他4个部件协调工作，共同完成任务。

(2) 程序存储思想: 信息都是存储在计算机存储器单元中，其形式为诸如0101010011…的二进制代码。根据CPU对单元内容的解释，这里又分为数据和指令: 数据是操作对象，指令是执行动作。在计算机中不仅可以存储要处理的数据，还可以存储进行数据处理的操作，也就是程序。

【例1.1】设计计算10＋20＝？的程序

计算10＋20＝？的程序可以分为如下的基本操作步骤：

（1）程序开始；

（2）从输入设备上读取第一个数（这里是10），存储在数据空间中编号为01的地址中；

（3）从输入设备上读取第二个数（这里是20），存储在数据空间中编号为02的地址中；

（4）把地址01中内容送入到运算器中；

（5）把地址02中内容送入到运算器中，进行加法运算；

（6）把运算器中的结果内容送到数据空间中编号为03的地址中；

（7）在输出设备上输出地址03中内容；

（8）程序结束。

在以上操作步骤中，每一步都可以转换为一条计算机的基本指令。从输入设备上读取数据要存放在数据空间的地址中，这样带来的好处是可以编写更为通用的程序。

从第3步开始，引用数据都是采用地址形式。当要计算30＋40＝？问题时，也可以使用上面程序，不需要重新编写。因为不管从输入设备上读入的数据是10也好，是30也好，它都存放在地址01中。当开始运算时，计算机会到地址01中取数，取到几就是几。所以，上面的指令序列实际上是解决$X+Y=?$这一类问题的通用程序。运算器把每一次运算结果存放在某个地址中，第5步把结果存放在编号为03的地址中。

所以，在设计程序时，必须要处理数据存储和操作流程。换句话讲，程序=数据结构+算法。

1.1.1 数据结构

从例1.1中可以看到，存储地址01、02、03是最简单的存储数据的结构。计算机程序要根据所要解决问题的复杂程度来合理安排数据的存储方式，这涉及数据结构的内容。例如，在10个数中查找最大值，则需要把10个数顺序地存储起来，这可以使用数组。在设计和编写计算机程序时，要根据解决问题的复杂程度合理使用诸如线性表、栈、队列、树等实用的数据结构。

数据结构是一门研究在计算机程序设计中操作对象以及它们之间存储关系和操作等内容的学科。关于数据结构不仅要考虑数据的数学性质，还必须考虑它的实现机制（存储和运行机制）。数据结构是学习信息管理、计算机开发与应用等知识的核心内容，我们在介绍Java程序设计时会适当地结合示例程序进行相关介绍。关于数据结构的系统学习，不是本书的重点，但是它是非常重要的。

1.1.2 算法

当有了数据存储之后，就要考虑按照何种流程来执行何种操作。

【例1.2】设计计算1＋2＋3＋…＋100的程序

这是一道小学的算术题。聪明的高斯想出了一个巧妙方法：1＋100是101，2＋99也是101，即从两边到中间依次相加，那么1＋2＋3＋…＋100就可以得到50个101，也就是说累

加之和应该是$50\times101=5050$。高斯的算法比较巧妙，他利用了这些数之间的规律从而避免了复杂的运算。

那么计算机程序应如何处理这个问题呢？高斯算法虽然很好，但其计算特征是：高斯的大脑加上纸上的计算过程。我们知道，计算机只能按人给它的指令来运行。从这一点讲，计算机是非常“笨”的。但是一旦给了它指令，它就可以准确无误地执行，不会疲劳，而且运行速度非常快，这一点又是人类大脑无法比拟的。因此我们可以把人类的聪明才智和计算机的刻苦耐劳、高速准确的特点结合起来，提供解决现实生活中仅靠人类大脑无法或很难解决的问题的解决方案。我们要把高斯大脑中的内容完全形式化地交给或者展现给计算机，也就是说，要把计算过程全部放在纸上。

对于计算机来说，算法的通用性更重要。一旦编制了一个程序之后，它应该能够解决一类问题，而不是一个问题。因此，计算机解决依次累加问题的算法是采用一个数一个数相加的方式，这样可以很容易扩展单个数值和所参加计算的数的个数。例如，参加运算的数值可以是整数、小数，运算个数可以任意指定。使用计算机解决问题的关键在于告诉计算机要做的事情，当然，只是简单地说明问题，计算机是不能理解的，必须明确每一步应该怎样做，即给计算机一个明确的算法。在解决$1+2+3+\cdots+100$的问题时，可以给出下列指令序列：

（1）算法开始；

（2）设置S为存储计算结果的存储单元，初始值为0；

（3）把S和1相加，所得的和记为S；

（4）将S与2相加，所得的和仍记为S；

（5）将S与3相加，所得的和仍记为S；

…

…

（101）将S与99相加，所得的和仍记为S；

（102）将S与100相加，所得的和仍记为S；

（103）打印S的值，S即为累加之和；

（104）算法结束。

S即为我们所求的结果。请大家留意，这里表述存储结果时已经采用了变量符号S，而不是具体的存储单元编号，显然这进行了一次抽象。对于人们来讲，我们只要在逻辑意义上说明存储结果的变量就可以了。但是计算机还是关注计算结果究竟存储在哪个具体地址单元中，变量符号与存储地址单元之间存在着对应关系。这种对应关系由计算机系统软件来处理。这样可以把固定的、成熟的处理方法累积和封装起来，让人们不用关心具体的实现，而把注意力始终关注到那些需要人们判断和解决的问题上。

当然这个方法在计算机上是可以实现的。但是可以看出，因为是100个数相加，这个算法有100条指令。那么如果有10000个数相加，难道要给计算机10000条指令？显然，这不是一个好的算法，因为它在人们对计算机布置任务的环节上太繁琐。

从前面算法中可以看出，它虽然有104条操作，但从第3步至第102步共计99条操作是相同的，都是两个数相加，只是操作对象是不同的数。这时，可以想到只给出一条操作，只

要它能根据不同的数来重复执行加法操作就可以了，不必写上99条操作。

下面给出改进的算法：

（1）算法开始；

（2）用S表示累加之和，开始时，$S=0$；

（3）用N表示要累加的数，开始时，$N=1$（从1开始累加）；

（4）做$S \leftarrow S+N$（把S和N相加，把两个数的和赋值给S）；

（5）$N \leftarrow N+1$（N变成下一个要累加的数）；

（6）如果$N \leqslant 100$，执行控制转向语句（4），否则执行控制转向语句（7）；

（7）打印结果S的值；

（8）算法结束。

从上面算法可以看出第（4）、（5）和（6）步操作是循环执行的，仅需要6步操作就可以满足要求，完成100个数的累加。算法中S表示前面所有数的累加之和，N表示当前正要累加的数。第（4）步把这两个数的相加结果再赋值给S，完成之后，S依然表示前面所有数的累加之和。第（5）步把N加1，表示下一个要累加的数。在第（6）步中，如果N小于等于100，说明还没有累加完，转移到第（4）步继续累加，否则说明已经加到100，达到要求，打印出结果值，算法结束。

在这个算法中，变量S和N的值都是不断变化的，在每次循环中以一个新的值代替它原来的值，然后再以新值作为下一次运算的基础。

算法不仅能解决诸如数学的计算问题，同时也要能够解决实际应用的非计算问题。

【例1.3】汉诺塔（Hannio）问题，如图1-1所示

所谓汉诺塔是给定三根针（分别为A、B和C针）和n个大小不同的、中间有圆洞的盘。这些盘可以穿针而堆放形成塔。设n个盘按照从小到大的次序堆放在A针上。任务是将这n个盘从A针移到C针，并且保持原来的堆放次序。完成这个任务有下列限制条件：

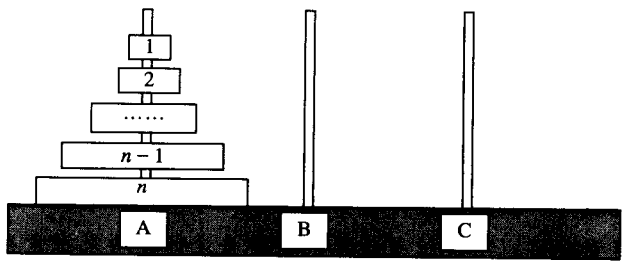

图1-1 汉诺塔

（1）每一次移动只允许严格地把一个盘从一根针移到另一根针；

（2）任何一个盘在任何时刻不允许堆放在任何比它小的盘上；

（3）B针可以作为辅助的、中间过渡的针。

请给出完成这个任务的算法。

设计算法的关键是判断盘数n。如果n是1，则直接把1号盘从A针移到C针就可以了。如果n大于1，则把待解决的问题（把n张盘从A针借助B针移到C针，如图1-2a所示）分解为3个子问题：

（1）首先，将$n-1$张盘从A针借助C针移到B针，如图1-2b所示；

（2）然后，再把第n号盘从A针直接移到C针，如图1-2c所示；

（3）最后，将$n-1$张盘从B针借助A针移到C针，如图1-2d所示。

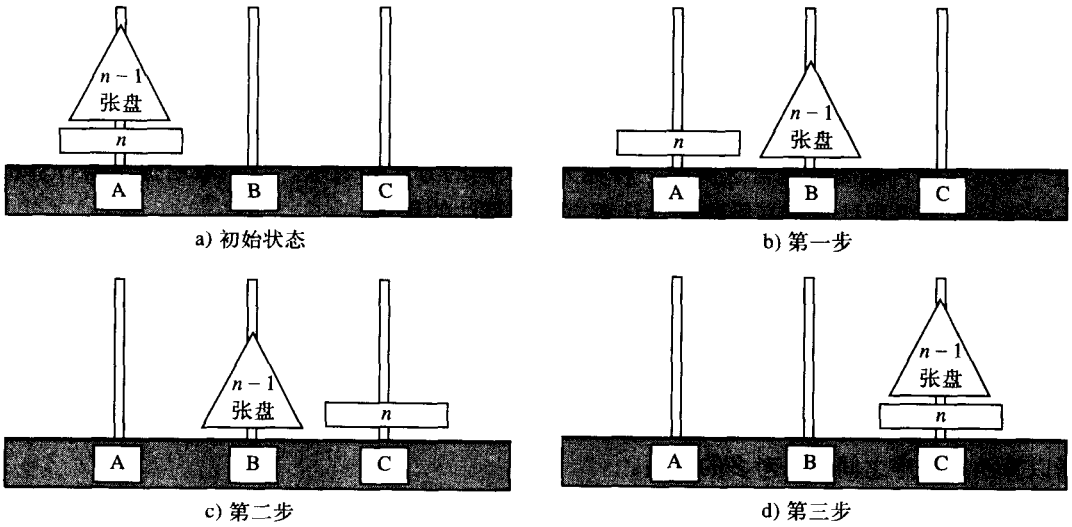

图1-2 汉诺塔的分解操作

特别注意：第n号盘与n张盘的含义是不同的。以上所说明的算法很容易被人们理解，但是它更加抽象了，换句话说，计算机面对这样的算法不知道该如何操作。

这里再来形式化地说明这个算法。设计一个方法moveto(n, from, to, middle)，其中n是表示要移动的盘数，from表示源针，to表示目的针，middle表示中间过渡的借助针。则moveto(n, from, to, middle)的处理过程如下：

（1）如果n＝1，则执行下列步骤：

a) 输出“请把第n号盘从from针移到to针”的信息；

b) 方法结束。

（2）如果n＞1，则执行下列步骤：

a) moveto(n-1, from, middle, to);

b) 输出“请把第n号盘从from针移到to针”的信息；

c) moveto(n-1, middle, to, from)。

moveto()关键采用了递归方法，每调用一次moveto()，则n值减1，总能够减到1，也就是说，算法能够结束。如果计算机软件系统支持递归功能，则人们不再需要设计详细步骤了，因为人们的想法计算机系统完全理解并且能够付诸实施了。如果计算机软件系统不支持递归功能，则需要设计处理递归的数据结构和操作。这需要进一步细化这个算法，直到所分解的每一步操作都能被计算机理解和执行，这才能说完成了算法设计。目前大多数计算机程序设计语言都支持递归功能，所以我们可以把moveto()交给计算机处理，输出所需要的结果。关于汉诺塔的递归和非递归程序在本书后面会详细讨论。

所以说，算法是完成特定任务的具体步骤集合的一种说明，也就是一组可执行操作的序列。算法具有下列特性：

（1）有穷性：一个算法应当在执行有穷步之后结束，不应该包含无终止的循环，即死循环。

（2）确定性：算法中的每一个步骤，必须是确切定义的，而不应当含混不清或模棱两可。

（3）具有零个或多个输入参数，即在算法开始执行前最初给出的值。

（4）算法执行后有一个或多个输出参数。例如求1到1000之间自然数的累加之和，则最后得到的和即为输出参数值。

（5）可执行性：算法中的每一步都是能够准确运行的。

一个算法可以采用自然语言的方式表示，也可以采用流程图的方式来表示，还可以采用介于自然语言和高级语言之间的一种称为伪代码的方式来表示。当然最后提交给计算机系统运行的是采用计算机高级程序设计语言实现算法的程序。

程序设计的关键在于设计出一个好的算法。这里的算法是一个广义的概念，不单指计算一道数学题，而是解决任何一个问题的步骤。一个好的算法应当能够获得正确的结果，容易阅读理解，可读性好，计算机执行时具有较高的效率（指得到结果所需的时间较少和占用计算机内存少）。

算法是独立于语言的。当设计出算法之后，就可根据它，采用某种高级语言编写程序。所以算法和程序之间是一对多的关系。

1.1.3 高级程序设计语言

每个国家、每个民族都有自己的语言和文字，作为人们之间交流思想的工具，这些语言称为自然语言，例如中文、英文等。人们要与计算机交流信息也需要解决“语言”问题，这就是计算机的程序设计语言。

计算机程序设计语言是人们为了向计算机准确描述实际情况而设计的，它经历了机器语言、汇编语言、高级程序设计语言的发展历史。机器语言使用指令代码来编写。汇编语言采用人们便于记忆和理解的助记符来表达机器指令，这只是便于人们更好地理解每条机器指令的含义，每条汇编语句与机器指令是一一对应的。高级计算机程序设计语言独立于机器，采用不依赖于机器具体指令的逻辑形式表达执行过程的高级算法语言。由此看来，程序是一种计算机语言指令的有序集合，它规定了计算机应当按怎样的步骤和规则来执行操作，以完成某项具体的任务。换言之，它是能被计算机理解并执行的步骤集合，其功能是用来解决某个实际问题，实现人们想要达到的目的。

因为不同规模和型号的计算机系统的指令是不同的，为了编制$X+Y=?$的程序，要熟悉每一种机器指令，这样的设计相当麻烦。随着计算机的应用日益广泛地渗透到各个学科和技术领域，60年代开发了许多不同风格、为不同对象服务的程序设计语言。例如，把上面的指令序列写成如下语句：

```
read A
read B
X=A+B
print X
end
```

显然这个语句序列比起前面的指令序列更适合人们理解，它比较定性地说明了两数相加的过程。对于计算机来讲，前面的指令序列操作更明确，比较定量地说明两数相加的具

体操作步骤。在这个语句序列和指令序列之间需要翻译或转换，这个工作是由计算机的编译程序来完成的，如图1-3所示。

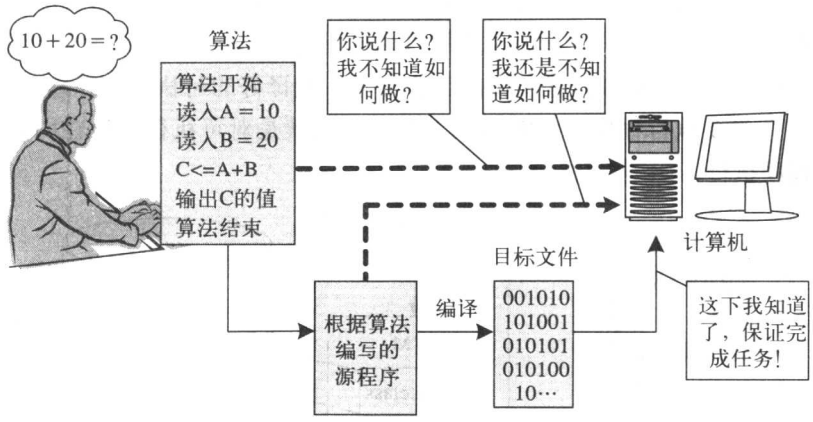

图1-3 高级程序设计语言以及机器指令与计算机间的关系

在图1-3中，我们可以看出：算法是抽象、枚举和归纳的，这是人们进行思维的一般方法。根据算法编写的源程序也是面向人们理解的，计算机看不懂。只有通过编译程序生成的目标代码（也称为机器代码）才能让计算机明白人们的意思，完成所交给的任务。

在计算机发展的早期，人们编写程序都是使用“0”和“1”两个代码来进行的，显然编写出来的程序，计算机会立即执行，而人们为了编写这些繁琐枯燥的二进制代码需要花费大量的精力，而且会频繁出现错误。随着计算机技术的发展，人们意识到：能否使用一种人们容易理解而又能形式化地由计算机系统转换到二进制形式的语言，这就是高级程序设计语言。随着高级程序设计语言的出现，人们可以把注意力集中到解决新问题的算法上，而具体转换成计算机系统“明白”的二进制程序和数据的过程就由编译程序全权代劳了，如图1-4所示。

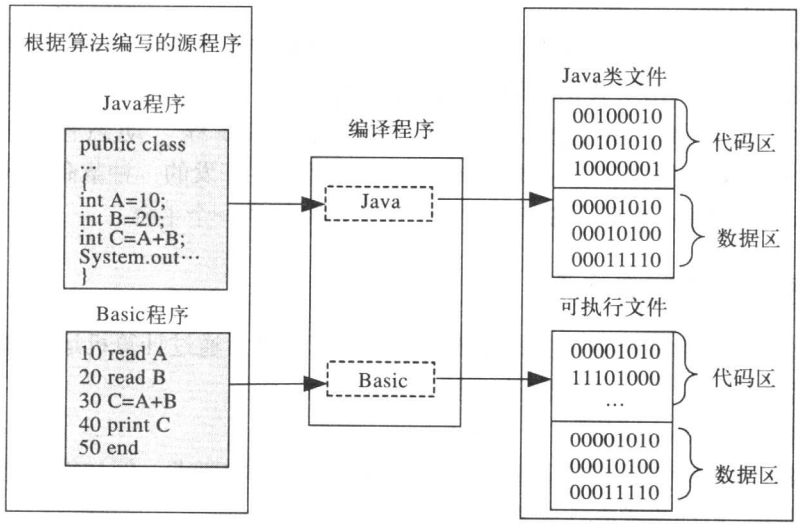

图1-4 编译过程

图1-4中列举出两个源程序：一个是Java源程序，一个是Basic源程序。它们都是按照同一个算法编制出来的，尽管语句形式和序列不一样，但都是解决$X+Y=?$这一类问题的程序。每种程序设计语言的编译程序是不同的，Java语言的编译程序是javac。编译程序所生成的目标代码也是不一样的，javac所生成的目标代码是Java类文件，即文件后缀名为.class。

具体到上面的例子，编译程序所做的工作如下：编译好的可执行文件存储在诸如硬盘、软盘的外存上，当要执行程序时，需要调入内存，也就是对可执行文件在内存中进行空间分配，启动程序执行，如图1-5所示。

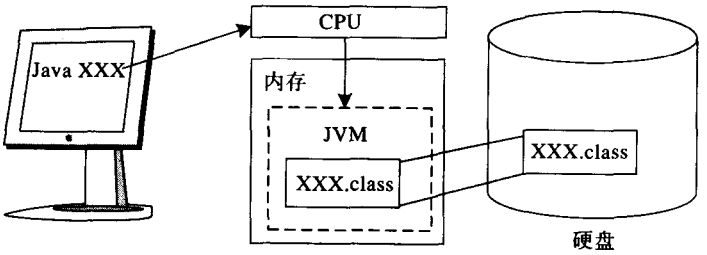

图1-5 程序的执行过程

在图1-5中，当用户从键盘并通过屏幕显示调用可执行文件时，要牢记可执行文件存储在硬盘上，图中显示为XXX.class。任何程序要运行，必须首先调用到内存中，得到执行资源之后才能执行。

计算机程序设计的发展就像滚雪球一样。从计算机裸机开始，增加机器指令或汇编语言，直到现在有了更适合人们习惯的程序设计开发平台。只要我们熟悉了机器指令，就可以指挥计算机按照人们的意愿来运行。机器指令的繁琐带来了容易出错，并且把人们精力限制在指令的汪洋大海中，而丢失了解决领域问题的方向。所以，人们需要使用适合人们理解的、更侧重于逻辑操作的高级程序设计语言。当前世界上已经有数百种程序设计语言，较为普遍使用的就有10多种。例如，COBOL、FORTRAN、BASIC、PASCAL、C/C++、VB、Delphi、Java等。

目前较为适合人们思维逻辑的语言应该是面向对象的程序设计语言。Java是一种面向对象的程序设计语言。由于其正确的设计目标（编写一次，到处运行）以及所提供的简洁性、分布性、坚固性、安全性、跨平台、高效、多线程、动态性等特征，随着Internet/Intranet企业应用的飞速发展，Java语言已成为软件开发的一种革命技术。从某种意义上讲，Java不仅仅是一种程序设计语言，而是管理资源的一个平台。

1.1.4 程序设计的基本步骤

程序设计应当包括从接受任务、分析问题开始，到最后通过计算机运行得到正确结果的全过程。程序设计一般包括以下6个步骤：

（1）明确所需解决的问题。

必须在接受任务时就确切地弄清楚问题的性质、任务和要求。例如给出什么数据，要求得到什么结果，打印格式有什么要求等等。

（2）分析问题、构造模型。