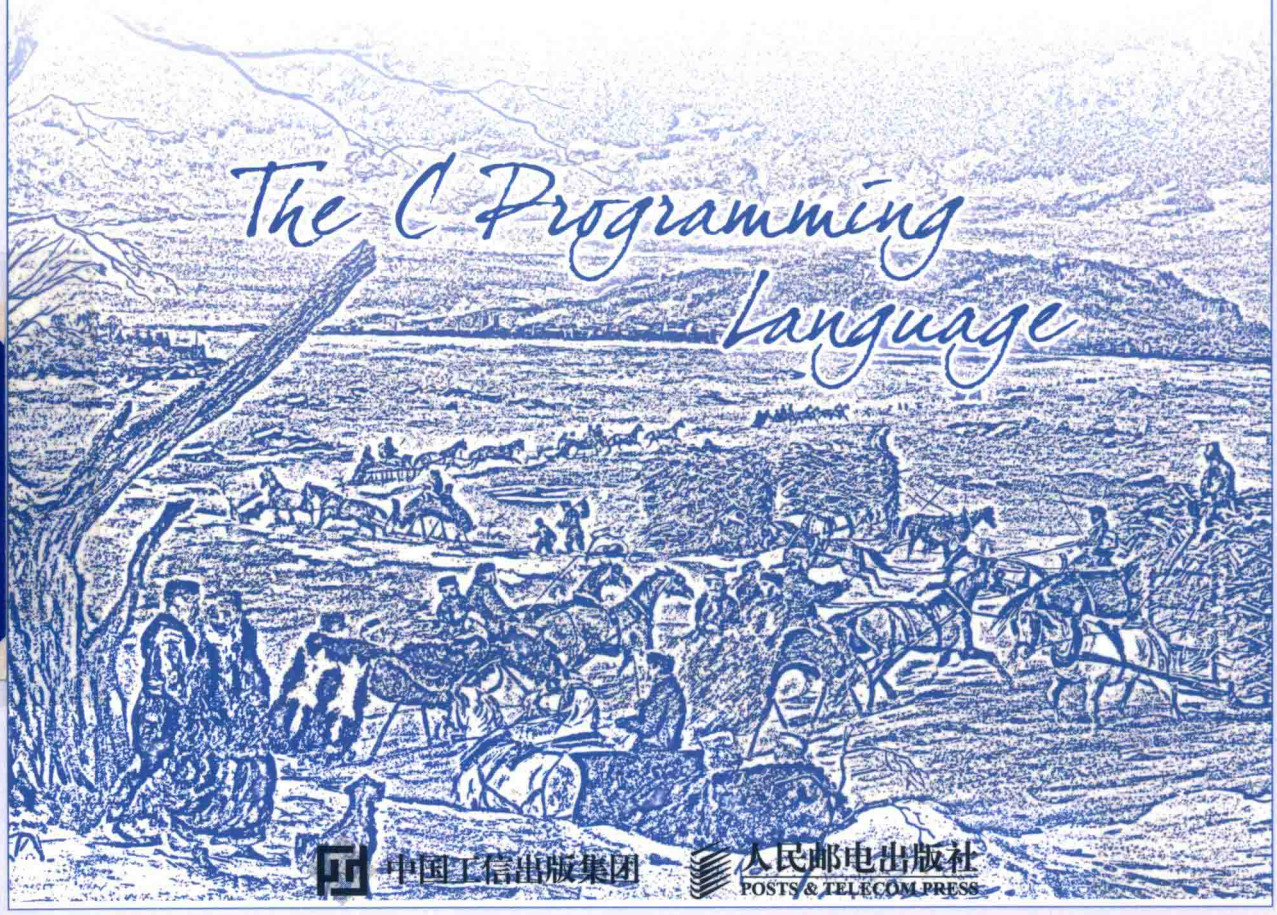
普通高等教育
软件工程
“十三五”规划教材

13th Five-Year Plan Textbooks
of Software Engineering

工业和信息化普通高等教育
“十三五”规划教材

C语言程序设计

杨崇艳 ◎ 主编
王幸民 王园宇 王星魁 ◎ 副主编



中国工信出版集团
人民邮电出版社
POSTS & TELECOM PRESS

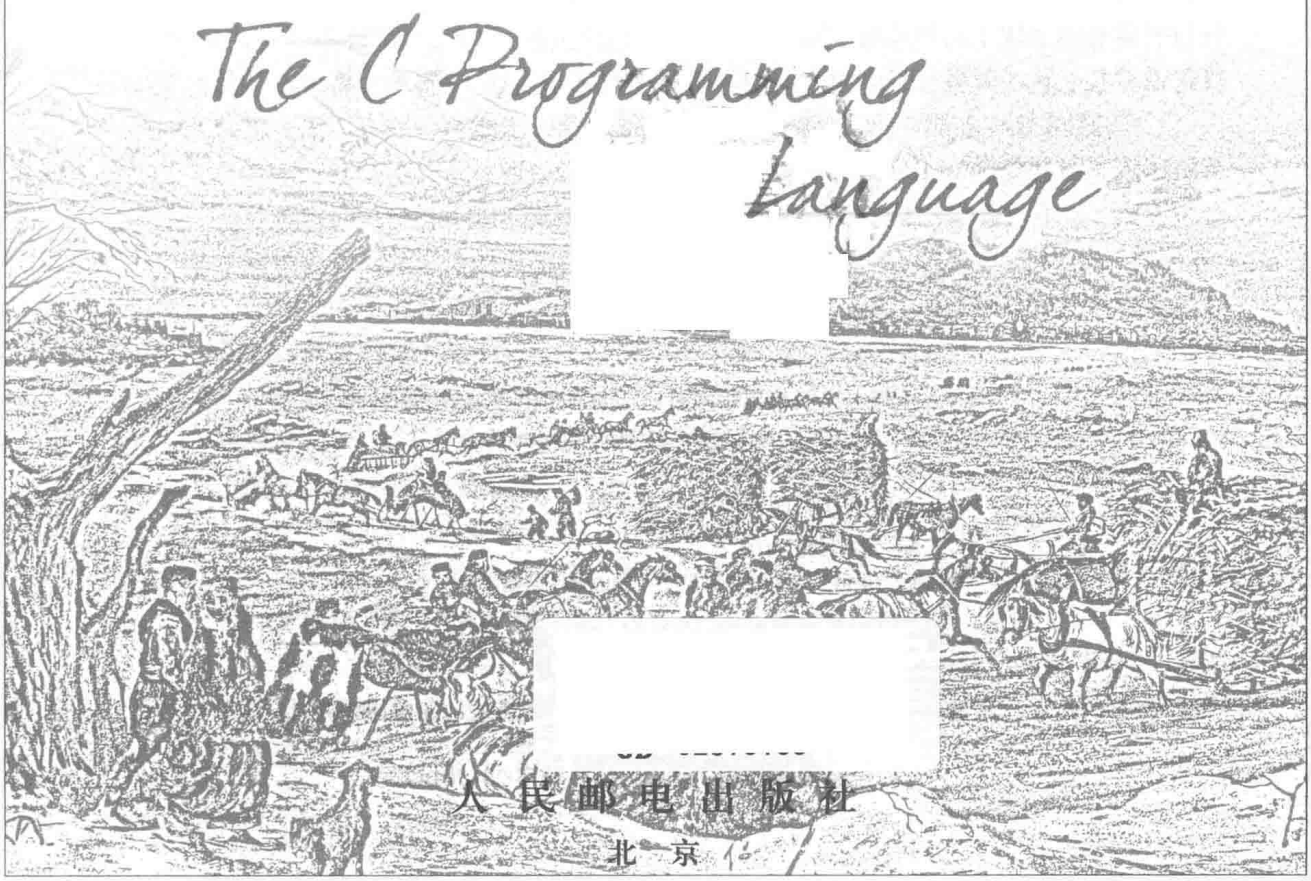
普通高等教育
软件工程
“十三五”规划教材

13th Five-Year Plan Textbooks
of Software Engineering

工业和信息化普通高等教育
“十三五”规划教材

C语言程序设计

杨崇艳 ◎ 主编
王幸民 王园宇 王星魁 ◎ 副主编



人民邮电出版社
北京

图书在版编目（CIP）数据

C语言程序设计 / 杨崇艳主编. -- 北京 : 人民邮电出版社, 2019.3（2019.4重印）
普通高等教育软件工程“十三五”规划教材
ISBN 978-7-115-50639-9

Ⅰ. ①C… Ⅱ. ①杨… Ⅲ. ①C语言－程序设计－高等学校－教材 Ⅳ. ①TP312.8

中国版本图书馆CIP数据核字(2019)第029246号

内 容 提 要

本书详细介绍C语言及其程序设计方法。全书共12章，主要内容包括：概述、C语言程序的数据描述及数据运算、顺序结构程序设计、选择结构程序设计、循环结构程序设计、数组、指针、函数和变量的存储类型、结构体与共用体、程序编译预处理、位运算、文件。本书展示了C语言灵活、高效的编程方法和在实践中的应用，努力做到理论与实践相结合。为了帮助读者学习，每章设有小结和习题。本书配有PPT、源代码等教学资源。

本书可作为高等学校学生学习C语言程序设计课程的教材，也可作为广大计算机程序设计人员和计算机程序设计爱好者的参考书。

◆ 主　　编　杨崇艳
副 主 编　王幸民　王园宇　王星魁
责任编辑　邹文波
责任印制　陈　犇

◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路11号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京捷迅佳彩印刷有限公司印刷

◆ 开本：787×1092　1/16
印张：19.75　　2019年3月第1版
字数：534千字　　2019年4月北京第2次印刷

定价：55.00元

读者服务热线：(010)81055256　印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京东工商广登字20170147号

前言

随着经济全球化、社会信息化时代的到来，当代大学生不但要会使用计算机获取专业领域知识，还要会使用计算机进行编程，利用信息技术进行复杂专业问题求解。

C 语言是一种被广泛使用的程序设计语言。它具有结构简单，功能强，数据类型丰富，表达能力强，运算灵活方便等特点。用它编写的程序，具有速度快、效率高、代码紧凑、可移植性好等优点。C 语言支持自顶向下、逐步求精的程序设计技术，其函数式结构为实现程序的模块化提供了强有力的保障。C 语言既具有高级语言的特性，又具有直接操控计算机硬件的能力，具有较强的系统处理能力。C 语言能够有效地用来编制各种系统软件和应用软件。

本书从程序设计与 C 语言之间关系出发，针对学生在学习 C 语言过程中的茫然，帮助学生首先树立对程序设计的整体认识，然后再进行有目的的 C 语言学习。

全书共 12 章，将 C 语言程序设计分成 3 个循序渐进的部分。

第一部分是入门基础，由第 1～3 章组成，主要介绍程序设计与 C 语言的基本知识，并引入算法的概念，使读者建立起程序设计的概念。这部分的内容奠定了 C 语言程序设计的基础。通过对这部分内容的学习，读者可以设计由表达式语句组成的简单程序。

第二部分是程序设计的基本结构，由第 4 章、第 5 章组成，主要介绍简单程序设计方法、简单算法的设计和用流程图来表达算法，以及两种常用的程序语句结构（分支结构和循环结构）。通过对这部分内容的学习，读者可以了解结构化程序设计的基本思想，掌握 C 语言程序的 3 种基本结构，学会设计简单的算法并能依据算法思想编写程序，掌握用 C 语言程序设计的思路来思考问题和解决问题。

第三部分是模块化程序设计的结构，由第 6～12 章组成，主要介绍利用数组批量处理数据，利用函数实现模块化程序设计。通过对这部分内容的学习，读者可以了解模块化程序设计的思想和方法，并进一步通过指针、结构体、预编译、位运算、文件等全面了解和掌握 C 语言程序设计的手段和方法，培养和锻炼开放性地思考问题和解决问题的能力。

针对教学过程中，预期教学目标与最终效果差距大，教师感觉难教，学生感觉难学、难理解，学了也不会编程的情况，我们在编写本书的过程中，从内容体系上做了精心的考虑，以求收到较好的效果。

全书体系合理，案例经典，以程序为核心，由浅入深，启发性强。

为了巩固所学的理论知识，本书每章都附有习题，以帮助读者理解基本概念，进一步熟练掌握 C 语言的基本思想和基本语句，提高程序设计能力。

另外，我们还编写了与本书配套的《C 语言程序设计习题与实验指导》。读者在学习过程中参考该书进行实践练习，有助于逐步掌握 C 语言程序设计的基本方法和基本技能，有助于提高解决问题的能力。

本书的建议学时数为 64 学时，其中课堂教学时数为 32 学时，上机实验时数为 32 学时，教师在实际教学中可以根据具体情况对书中内容进行取舍，适当减少或增加学时数。C 语言程序设计是一门实践性很强的课程，课堂教学必须与实践相结合，实践的作用更大一些。因此，广大读者必须通过大量的编程训练，在实践中掌握语言知识，培养程序设计的基本能力，并逐步理解和掌握程序设计的思想和方法。希望广大读者在学习语言的过程中做到以下两点。

（1）学好语法知识。它们是编写计算机程序的基础。

（2）多上机、多思考、多模仿。只有这样，才能提高自己的编程能力，为将来的学习和工作打下坚实的基础。

本书是程序设计的基础教材，不但适合初学者使用，对专业人员也有一定的参考价值。本书可以作为普通高等院校计算机专业及理工类各专业本科、专升本的教材，也可作为计算机水平考试培训、各类成人教育院校程序设计课程的教材，以及作为计算机应用工作者和工程技术人员的参考书。

本书由杨崇艳担任主编并统稿，王幸民、王园宇、王星魁担任副主编。其中，第 1 章由王幸民编写，第 2 章由王娜编写，第 3 章由李月华编写，第 4 章由王颖编写，第 5 章由杨丽凤编写，第 6 章由王园宇编写，第 7 章由杨崇艳编写，第 8 章由王星魁编写，第 9 章由贾晓华编写，第 10 章由任少斌编写，第 11 章由刘永红编写，第 12 章由雷红编写，附录 1～附录 4 由张海超编写。相洁教授在百忙之中抽出时间对本书进行了审阅。在此对他们及所有为本书的出版付出了辛勤劳动的同志表示衷心的感谢。

此外，本书的出版还得到了山西省教育科学“十三五”规划“1331 工程”研究专项课题（编号：ZX-18017）的支持。

由于编者水平有限，加之时间仓促，书中难免存在不妥与疏漏之处，敬请广大读者批评指正。

编　者

2019 年 1 月

目 录

第1章 概述

你可能使用QQ或微信进行网络聊天，可能知道神舟十一号载人航天器在距地面393km的轨道与天宫二号对接，你也可能知道石油地质勘探使用大型计算机进行模拟运算。那么，你是否曾想过，是什么使计算机或智能终端等硬件设备有如此丰富多彩的功能？是什么样的“大脑”能实现如此精准的控制？是软件，是用某种计算机语言编写的程序实现了上述功能。

本章首先引入计算机的工作机制，包括硬件和软件两部分；然后介绍程序与程序设计、算法与算法描述和结构化程序设计思想的相关知识，使读者建立起对程序、程序设计、算法与算法描述，以及采用结构化程序设计方法实现问题的求解过程的基本认知；最后介绍C语言的一些基础知识，包括C语言的发展及特点、C语言程序的结构及执行过程，通过实例介绍如何用计算机解决实际问题。

本章重点内容如下。

（1）程序与程序设计。

（2）算法与算法描述。

（3）C语言程序的结构及执行过程。

1.1 计算机的工作机制

计算机程序不同于其他程序（如毕业典礼程序），它是由计算机来执行的。计算机程序的编制（程序设计）通常要考虑到计算机解决问题的方式和特点。因此，要进行程序设计，就有必要对计算机的工作机制有一定的了解。

1.1.1 硬件结构

虽然现在计算机硬件所提供的计算能力与早期的计算机相比有了很大的提高，但是，目前大部分计算机基本上采用的还是冯·诺依曼体系结构，即存储程序式结构。冯·诺依曼计算机的硬件构成如图1-1所示。

1. 中央处理器

中央处理器（简称CPU）是计算机的核心部件，它用于执行计算机指令，以完成计算任务。CPU由控制器、运算器以及寄存器等构成。控制器负责从内存中取指令并根据指令发出控制信号以引起其他部件的动作。运算器执行运算指令所规定的运算。寄存器主要用于记录下一条指令的内存地址、当前指令的执行状态以及暂时保存指令的计算结果，其作用主要是减少访问内存的次

数，提高指令的执行效率。

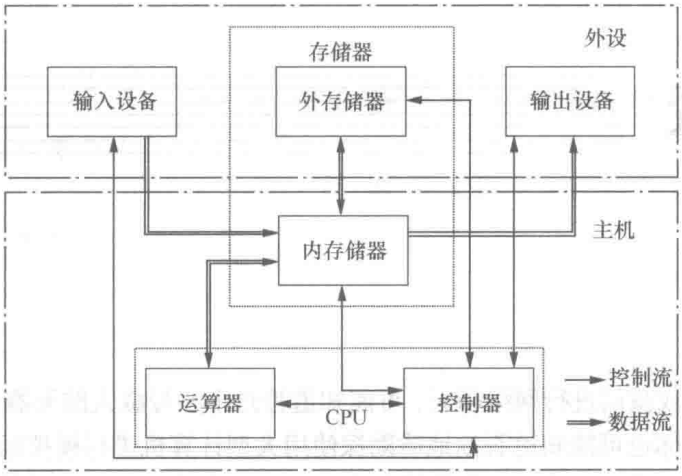


图 1-1　典型的计算机硬件组成

2. 内存储器

内存储器（简称内存）用于存储计算机程序。内存由许多存储单元构成，存储单元的大小视计算机的规格而定。对于微型计算机而言，内存存储单元的大小一般为一个字节，每个存储单元都有一个地址，对存储单元的访问是通过其地址来进行的。与 CPU 内的寄存器相比，内存容量要大得多，但访问内存单元所花费的时间要比访问寄存器多得多。

3. 外部设备

外部设备（简称外设）提供了计算机与外界的接口，主要用于计算机的输入/输出以及为计算机提供大容量的信息存储。外设包括输入设备、输出设备以及外存储器（简称为外存）。键盘和鼠标等属于输入设备，显示器和打印机等属于输出设备。

外存是大容量的低速存储部件（与内存相比），用于永久性地存储程序、数据以及各种文档等信息。外存包括硬盘、光盘、磁带等。存储在外存中的信息通常以文件形式进行组织和访问。外存与内存除了在容量和速度上不同外，它们的另一个区别在于：内存中存储的是正在运行的程序和正在使用的数据，而外存中存储的则是大量的、并非正在使用的程序和数据。

冯·诺依曼计算机的工作模型是：待执行的程序从外存装入到内存中，CPU 从内存中逐条地取出程序中的指令执行；程序执行中所需要的数据从内存或外设中获得，程序执行中产生的中间结果保存在内存中，程序的执行结果通过外设输出。上述计算过程的本质是通过不断地改变程序的状态来实现计算，程序的状态由内存单元的数据构成，状态的转换由指令来实现。

CPU 所能执行的指令如下。

（1）算术指令。实现加、减、乘、除等运算。

（2）比较指令。比较两个操作数的大小。

（3）数据传输指令。实现 CPU 的寄存器、内存以及外设之间的数据传输。

（4）执行流程控制指令。用于确定下一条指令的内存地址，包括转移、循环以及子程序调用/返回等指令。通常情况下，CPU 从某个内存地址开始依次读取指令来执行，执行流程控制指令可以用来改变程序顺序执行的行为。

由于构成计算机的各个部件存在速度上的差别，快速部件往往要花费大量的时间等待慢速部件的操作，因此，在冯·诺依曼计算机中存在着几个影响程序执行效率的瓶颈，它主要体现在 CPU

与内存之间以及内存与输入/输出设备之间的数据传输。现在的计算机中往往利用程序执行和对数据访问所具有的局部性特征，通过缓存机制来解决部件之间速度不匹配问题，从而提高计算机的整体性能。缓存中存储的是近期用过的、今后可能还要用到的一些内容。例如，目前的计算机大都在 CPU 中为内存提供高速缓存，在内存中为外存提供高速缓存。

1.1.2　软件系统

计算机硬件只是提供了执行存储在内存中指令的能力，而执行的指令是需要人来提供的。也就是说，计算机硬件为计算机提供了物质基础，但它必须通过计算机软件来发挥作用。计算机软件是计算机系统中的程序以及相关文档。程序是对计算任务的处理对象（数据）与处理规则（算法）的描述；文档是为了便于人理解程序所需要的资料说明，供程序开发与维护使用。

软件系统可以分为系统软件、支撑软件和应用软件。系统软件居于计算机系统中最靠近硬件的一级，它与具体的应用领域无关，其他软件一般要通过系统软件发挥作用，如操作系统就属于系统软件。支撑软件是指支持软件开发与维护的软件，一般供开发人员使用，如软件开发环境就是典型的支撑软件。应用软件是指用于特定领域的专用软件，如学生管理软件等。计算机软件的分类如图 1-2 所示。

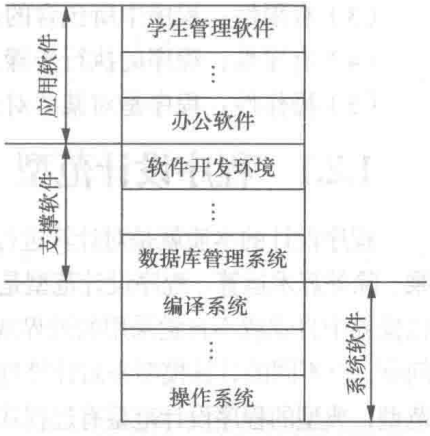


图 1-2　计算机软件分类

一个软件从无到有，一直到最后的消亡，通常要经历一个过程，这个过程称为软件的生存周期。软件生存周期分成若干个阶段：软件需求分析、软件设计、编程实现、测试及运行与维护。软件需求分析的主要任务是明确待实现的软件要解决什么问题，即做什么，给出软件的需求说明。软件设计是根据软件的需求说明给出抽象的解决方案（设计说明），它包括概要设计和详细设计。概要设计是指软件的整体结构设计，详细设计是指抽象的数据结构和算法描述。编程实现是指根据软件设计说明，采用某种程序设计语言编写程序。测试是对编好的程序进行测试，确认其是否满足所规定的需求。运行与维护是指使用软件并在使用过程中发现和改正程序中的错误。值得一提的是，在维护工作上的投入往往要占整个软件生存周期的很大一部分，这是因为只要软件在使用，就需要对它进行维护。

早期的软件开发工作主要花费在编程实现阶段，并且采用的是个体的小作坊开发模式。随着计算机应用领域的不断扩大和应用层次的不断加深，使得软件的规模不断扩大、软件的复杂度不断提高，早期的软件开发模式难以驾驭软件开发过程，程序的正确性难以保证，软件生成率急剧下降，出现“软件危机”。为了解决软件危机问题，软件工程概念应运而生，其主要思想是采用工程方法来开发软件。在软件工程中，软件开发工作的中心从实现阶段转移到软件需求分析、设计和维护阶段，并且强调对软件开发过程的管理和加强各个阶段的文档制作。方法和工具构成了软件工程的两大支柱，它们贯穿于软件开发过程，对软件工程思想提供具体的支持。

1.2　程序与程序设计

现代计算机具有计算速度快、计算精度高、自动化程度好以及通用性强等特点，在我们的生

活中扮演着越来越重要的角色，被广泛应用于数值计算、数据处理、自动控制、计算机辅助设计以及人工智能等众多的领域。然而计算机并不能自主地实现这些任务，其每一步的操作都是在计算机程序的指挥下完成的，可以说计算机程序就是计算机的灵魂。

“程序”一词来源于生活，通常指完成某项工作的一整套活动过程及活动方式。有了程序这个概念，人们就可以对一系列步骤的执行过程进行详实地描述。

同样的道理，计算机要正确地运行并且完成相应的任务，也需要按照计算机程序的安排去执行。在计算机当中，程序是指使计算机完成某一特定任务而编写的若干条指令的有序集合。计算机程序往往简称为程序，这是计算机系统中最基本的概念。

一个计算机程序具有如下性质。

（1）目的性：程序有明确的目的，在运行时能正确完成赋予它的功能。

（2）分步性：程序为完成其复杂的功能，由一系列计算机能执行的步骤组成。

（3）有限性：程序中所包含的步骤是有限的。

（4）有序性：程序的执行步骤是有序的，不能随意改变这些步骤的执行顺序。

（5）操作性：程序是对某些对象的操作，能完成有意义的功能。

1.2.1 程序设计范型

程序设计的本质就是对计算进行描述。这里所讲的计算是指广义上的计算，而不是简单的加、减、乘、除等算术运算。程序设计范型是计算机编程中的基本风格和典范模式，是编程者在其所创造的虚拟世界中自觉或不自觉采用的世界观和方法论。范型引导人们带着其特有的倾向和思路去分析和解决问题。以不同的计算模型来对计算进行描述就形成了不同的程序设计范型。目前存在若干种程序设计范型，典型的程序设计范型有过程式（面向过程）、对象式（面向对象）、函数式以及逻辑式等。

1. 过程式

过程式程序设计是一种以功能为中心、基于功能分解的程序设计范型。一个过程式程序由一些子程序构成，每个子程序对应一个子功能，它实现了功能抽象。子程序描述了一系列的操作，它是操作的封装体。过程式程序的执行过程体现为一系列子程序调用。在过程式程序中，数据处于附属地位，它独立于子程序，在子程序调用时作为参数传给子程序使用。著名的计算机科学家 Nicklaus Wirth 提出了如下的经典公式，刻画了过程式程序设计的本质特征。

程序=数据结构+算法

上述公式中的算法是指对数据的加工步骤的描述，而数据结构则是对算法所加工的对象的数据描述。早期的程序设计都采用了过程式程序设计范型，它与冯·诺依曼计算机模型直接对应。过程式程序设计对程序功能的描述比较清晰，所描述的计算过程容易理解。过程式程序设计不足之处在于：数据与操作分离，缺乏对数据的保护；功能会随着需求的改变而发生变化，而功能的变化往往会导致整个程序结构的变动，使得程序难以维护；子程序往往是针对某个应用而设计的，它们很难用于其他应用程序，导致程序难以复用。

2. 对象式

对象式程序设计是一种以数据为中心、基于数据抽象的程序设计范型。对象式程序设计通常称为面向对象程序设计。一个面向对象的程序由一些对象构成，对象是由一些数据及可施加于这些数据上的操作所构成的封装体。对象的特征由相应的类来描述，一个类可以从其他的类继承。面向对象的程序的执行过程体现为各个对象之间相互发送和处理消息。面向对象程序可简单地表示成下面的公式：

程序=对象/类+对象/类+…

对象/类=数据+操作

在面向对象程序设计中，把数据和对数据的操作封装在一起，对数据的操作必须通过相应的对象来进行，从而加强了对数据的保护。对象是相对稳定的实体，由对象构成的程序能够适应软件需求的变化，易于维护。某个领域中的对象往往具有通用性，它们可以用于该领域类似的系统中，因此面向对象程序设计对软件复用有较好的支持。另外，面向对象程序设计范型是对问题领域活动的直接模拟，其中的对象往往对应着问题空间中的有形或无形的实体，它使得解题空间有自然的对应关系，从而有利于对大型复杂问题给出解决方案，使得程序容易设计、容易理解与容易维护。面向对象程序设计的不足之处在于：对程序的整体功能描述不明显；程序会包含较多冗余信息，这对小型应用系统有时不适合；程序效率有时不高。

3. 函数式与逻辑式

函数式程序设计是围绕函数及函数应用来进行的，它基于递归函数理论和 λ 演算（λ 演算即一套用于研究函数定义、函数应用和递归的形式系统），其中，函数也被作为值来看待。逻辑程序设计是把程序组织成一组事实和一组推理规则，它基于的是谓词演算。上述两种程序设计范型常用于人工智能领域的程序开发。

目前，使用较广泛的是过程式和对象式这两种程序设计范型。它们已成为现在的主流程序设计范型，适合于解决大部分的实际应用问题，已被广大的程序设计者所熟悉和采用。本教程只围绕过程式程序设计范型展开，介绍过程式程序设计的基本思想和技术。

1.2.2　程序设计语言

程序设计的结果必然要用一种能被计算机接受的语言表示出来，即编程实现。计算机程序设计语言是一个能完整、准确和规则地表达人们的意图，并用以指挥或控制计算机工作的“符号系统”。简单地说计算机程序设计语言是人与计算机进行信息通信的工具。

1. 程序设计语言分类

按照计算机语言的发展过程，程序设计语言可以分为机器语言、汇编语言和高级语言 3 大类。

（1）机器语言

机器语言是由 0、1 组成的机器指令的集合。机器语言可以被计算机直接执行。然而不同型号的计算机其机器指令是不能通用的，即按照某种计算机的机器指令编写的程序，不能在另一种计算机上执行，所以机器语言是一种面向机器的语言，也被称为低级语言。机器语言程序具有计算机能够直接识别、执行效率高的优点，但其具有书写难、记忆难、编程困难以及可读性差等缺点。目前，除了计算机生产厂家的专业人员外，绝大多数程序员已经不再去学习机器语言了。

【例 1-1】用机器语言程序实现“6+8”的运算。

```
10110000  00000110  ：将 6 送入累加器 AL 中
00000100  00001000  ：8 与累加器 AL 中的值相加，并将结果放在 AL 中
11110100            ：停机结束
```

（2）汇编语言

汇编语言克服了机器语言的一些缺点，采用助记符和符号地址来表示机器指令，因此也称作符号语言。在【例 1-2】中，用助记符“MOV”表示数据传送，代替了例 1-1 中的机器指令“10110000”；用助记符“ADD”表示加法运算，代替了例 1-1 中的机器指令“00000100”；用助记符“HLT”表

示停机结束，代替了【例 1-1】中的机器指令“11110100”，这样使程序的可读性有了很大的提高。

【例 1-2】用汇编语言程序实现“6+8”的运算。

```
MOV  AL, 06
ADD  AL, 08
HLT
```

汇编语言增加了程序的可读性，但其还是低级语言，它也是面向机器的语言。用汇编语言编写的程序不能被计算机直接识别和执行，必须要经过“汇编程序”（一种能把用汇编语言编写的程序翻译成机器语言程序的软件）将其转换成机器语言之后才能执行，这一过程称为汇编。由于汇编语言比机器语言可读性好、执行效率高，所以，许多系统软件的核心部分仍采用汇编语言编制。

（3）高级语言

高级语言是一种接近于自然语言的程序设计语言，它按照人们的语言习惯，使用日常用语、数学公式和符号，按照一定的语法规则来编写程序。

【例 1-3】用 C 语言程序实现“6+8”的运算。

```
#include<stdio.h>
int main( )                  /* 主函数 */
{   int a;                   /* 定义整型变量 a */
    a=6+8;                   /* 6+8 的结果赋给 a */
    printf("a=%d\n", a);     /* 显示结果 */
    return 0;
}
```

高级语言与自然语言更接近，而与硬件功能相分离（彻底脱离了具体的指令系统），编程者不必了解过多的计算机专业知识便可掌握和使用。高级语言的通用性强，兼容性好，便于移植。高级语言的产生，有力地推动了计算机软件产业的发展，进一步扩展了计算机的应用范围。

2. 程序翻译方式

使用高级语言编写的程序称为高级语言源程序，但计算机不能直接接受和执行源程序，必须通过“翻译程序”将其翻译成机器语言形式后才能执行。这种“翻译”通常有两种方式：编译方式和解释方式。

（1）编译方式

编译方式需要事先编好一个被称为“编译程序”的程序，将其放在计算机中。当高级语言源程序输入到计算机中时，编译程序便把源程序全部翻译成机器指令表示的目标程序；然后执行该目标程序，得到计算结果，如图 1-3 所示。

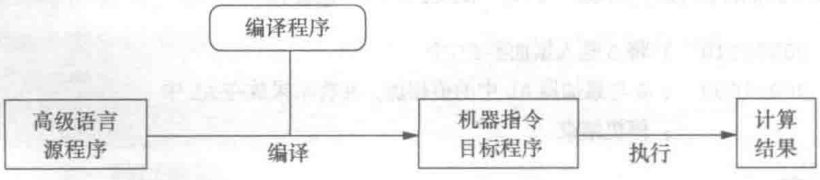


图 1-3　编译方式

（2）解释方式

解释方式需要事先编好一个被称为“解释程序”的程序，将其放在计算机中。当高级语言源

程序输入到计算机中时，解释程序将源程序的每一条语句逐句翻译，逐句执行，也就是边解释边执行，如图 1-4 所示。需要注意的是，解释方式并不产生目标程序。

现在用于编程的语言有很多，目前流行的 Python 语言属于解释方式，其简单易学，开发高效，可移植，可扩展，可嵌入；而 C 语言属于编译方式。目前，高级语言正朝着面向问题和面向对象的设计方向发展。相信未来的高级语言将会更加便于编程人员的使用。

图 1-4　解释方式

1.2.3　程序设计步骤

因为计算机的本质是程序的机器，所以只有通过对程序设计的学习才能更好地掌握和使用计算机。程序设计就是设计和编制程序的过程，是将实际问题用计算机方法解决的一个转化过程。尽管不同规模的程序因其复杂程度不同，设计步骤有所差异，但是一些基本步骤是相同的。

（1）问题分析：一般来说程序要解决的是一个具体的问题，所以程序设计人员需要具体问题具体分析，在把任务交给计算机处理之前必须对问题做出明确的分析与定义，确定问题的条件及期望的结果，找出解决问题的规律。

（2）算法设计：算法是解决问题的步骤及其描述，是程序设计的核心内容。算法是根据问题分析中的具体要求而设计的，是对问题处理过程的进一步细化。算法设计过程中，要求采用某种算法描述工具来表示程序运行的具体步骤。

（3）程序编码：编码就是使用计算机编程语言编写源程序代码的过程。在这个过程中，首先应当选择编程语言，然后用该语言来实现上一步骤中设计好的算法。应当注意，对于相同的算法，采用不同的编程语言会对程序的执行效率产生影响。

（4）调试运行：指在计算机上调试程序。调试运行程序有两个目的：一是消除由于疏忽而引起的语法错误或逻辑错误等；二是用各种可能的输入数据对程序进行测试，验证程序是否对各种合理的数据都能得到正确的结果，而对不合理的数据也能做适当的处理。

（5）文档编制：指整理并写出文字材料。文档包括程序说明文件和用户操作手册。程序说明文件记录了程序使用的算法、实现过程，用以保证程序的可读性和可维护性。用户操作手册则需要包含程序功能、运行环境、程序的安装与启动、基本参数的输入等内容。对于开发、维护周期较长的程序来说，适时地编制相应的文档显得尤其重要。

1.3　算法与算法描述

算法解决“做什么”和“怎么做”的问题，算法设计在程序设计中的地位非常重要。打个比方，厨师制作菜肴，需要有菜谱，菜谱上一般应有说明：①所用原料，指出为了做出顾客所指定的菜肴，应该使用哪些材料；②操作步骤，指出有了这些原料，应按照怎样的步骤进行加工，才能做出所需的菜肴。没有原料是无法加工成所需菜肴的，而对同一些原料可以加工出不同风味的菜肴。这里的“原料”相当于数据结构，“操作步骤”相当于算法。

本书由于篇幅所限，不可能将这些知识都详细讲解，感兴趣的读者可以深入学习相关的专

业知识。本节主要对算法的有关知识做初步介绍，以便为后面各章的学习奠定一些基础。

1.3.1 算法概念

算法是指为解决某个特定问题而采取的方法和步骤。算法是根据具体问题而设计的，问题不同，所设计的算法就不同。当然，即使对于同一个问题，也可以用不同的算法加以解决。我们这里所说的算法一般是指用计算机解决问题的方法和步骤，即计算机算法。

一个算法的设计具备如下特点。

（1）有穷性。它包含两个方面：一方面是指一个算法应在有限的操作步骤内完成，另一方面是指算法操作应在有限的时间范围内完成。

（2）确定性。算法中的每一个步骤都是确定的，即不能有二义性，这样才能确保对于同一个算法，相同的输入必然得出相同的执行结果。

（3）有零个或多个输入。输入是指算法所需要的外部信息。在计算机上实现的算法，是用来处理数据对象的，在大多数情况下这些数据对象需要通过输入来得到。

（4）有一个或多个输出。算法是有目的的操作，算法的目的是为了求解，这些解只有通过输出才能得到。没有输出的算法是没有意义的。

（5）有效性。算法中的每一个步骤都应当能有效地执行，并得到确定的结果。

算法有顺序结构、选择结构和循环结构这三种基本的控制结构。

【例 1-4】任意输入两个数，按降序（从大到小排列）输出到屏幕上。请给出解决问题的算法。

分析：将任意两个数按降序排列，首先要比较大小，如果前一个数小就交换并输出，否则按原顺序输出。

步骤 1：将输入的任意两个数赋给两个变量。

步骤 2：比较这两个变量中的数，如果前一个数小就交换，否则不交换。

步骤 3：将结果输出到屏幕上。

1.3.2 算法描述

对算法的描述有自然语言、流程图、N-S 图、伪代码等表示方法，下面分别进行介绍。

1. 自然语言

自然语言即人们日常生活中使用的语言。使用自然语言描述算法，通俗易懂，初学者容易掌握。【例 1-4】就是采用自然语言进行的描述。然而自然语言的含义有时会模糊，易发生歧义，描述文字显得冗长。这都容易造成理解上的偏差，因而不易直接转化为程序，所以用自然语言描述算法不太合适。通常除了一些很简单的问题外，一般不用自然语言表示算法。

2. 流程图

流程图是一种被广泛使用的算法描述方法，它用一些图框和流程线来表示各种类型的操作。常见的流程图符号如图 1-5 所示。流程图方法形象直观，易于理解，便于发现算法出现的错误，可直观地将算法转化为程序。

图 1-5 流程图符号

流程图描述的 3 种基本控制结构如图 1-6 所示。图 1-6（a）为顺序结构，只有 A、B 两个操