

全国计算机等级 考试二级教程



教育部考试中心

——C 语言程序设计 (2018年版)

高等教育出版社



全国计算机等级考试二级教程

——C 语言程序设计

(2018 年版)

Quanguo Jisuanji Dengji Kaoshi Erji Jiaocheng
——C Yuyan Chengxu Sheji

教育部考试中心

主编 田淑清

参编 周海燕 孙甲松

叶海建

高等教育出版社·北京

内容提要

本书根据教育部考试中心制订的《全国计算机等级考试二级 C 语言程序设计考试大纲(2018年版)》的要求而编写,是教育部考试中心组织编写的计算机等级考试系列教程之一。本书主要内容包括:程序设计基本概念,C程序设计的初步知识,顺序结构,选择结构,循环结构,字符型数据,函数,地址和指针,数组,字符串,对函数的进一步讨论,C语言中用户标识符的作用域和存储类,编译预处理和动态存储分配,结构体、共用体和用户定义类型,位运算,文件。

本书内容精练,结构合理,便于自学,对读者可能遇到的难点做了十分系统、清楚和详细的阐述,极大地减轻了读者学习 C 语言的困难,是应试人员考前必备的教材,也可作为普通高等学校的教材。

图书在版编目(C I P)数据

全国计算机等级考试二级教程: 2018 年版. C 语言程序设计 / 教育部考试中心编. --北京: 高等教育出版社,2017. 11
ISBN 978-7-04-048852-4

I. ①全… II. ①教… III. ①电子计算机-水平考试-教材②C 语言-程序设计-水平考试-教材 IV. ①TP3

中国版本图书馆 CIP 数据核字(2017)第 272489 号

策划编辑	何新权	责任编辑	何新权	封面设计	李小璐	版式设计	杜微言
责任校对	李大鹏	责任印制	耿 轩				

出版发行	高等教育出版社	网 址	http://www.hep.edu.cn
社 址	北京市西城区德外大街 4 号		http://www.hep.com.cn
邮政编码	100120	网上订购	http://www.hepmall.com.cn
印 刷	北京市鑫霸印务有限公司		http://www.hepmall.com
开 本	787mm×1092mm 1/16		http://www.hepmall.cn
印 张	18.25	版 次	2017 年 11 月第 1 版
字 数	450 千字	印 次	2017 年 11 月第 1 次印刷
购书热线	010-58581118	定 价	36.00 元
咨询电话	400-810-0598		

本书如有缺页、倒页、脱页等质量问题,请到所购图书销售部门联系调换
版权所有 侵权必究
物 料 号 48852-00

积极发展全国计算机等级考试 为培养计算机应用专门人才、促进信息 产业发展作出贡献

(序)

中国科协副主席 中国系统仿真学会理事长
第五届全国计算机等级考试委员会主任委员
赵沁平

当今,人类正在步入一个以智力资源的占有和配置,知识生产、分配和使用为最重要因素的知识经济时代,也就是小平同志提出的“科学技术是第一生产力”的时代。世界各国的竞争已成为以经济为基础、以科技(特别是高科技)为先导的综合国力的竞争。在高科技中,信息科学技术是知识高度密集、学科高度综合、具有科学与技术融合特征的学科。它直接渗透到经济、文化和社会的各个领域,迅速改变着人们的工作、生活和社会的结构,是当代发展知识经济的支柱之一。

在信息科学技术中,计算机硬件及通信设施是载体,计算机软件是核心。软件是人类知识的固化,是知识经济的基本表征,软件已成为信息时代的新型“物理设施”。人类抽象的经验、知识正逐步由软件予以精确地体现。在信息时代,软件是信息化的核心,国民经济和国防建设、社会发展、人民生活都离不开软件,软件无处不在。软件产业是增长快速的朝阳产业,是具有高附加值、高投入高产出、无污染、低能耗的绿色产业。软件产业的发展将推动知识经济的进程,促进从注重量的增长向注重质的提高方向发展。软件产业是关系到国家经济安全和文化安全,体现国家综合实力,决定 21 世纪国际竞争地位的战略性的产业。

为了适应知识经济发展的需要,大力促进信息产业的发展,需要在全民中普及计算机的基本知识,培养一批又一批能熟练运用计算机和软件技术的各行各业的应用型人才。

1994 年,国家教委(现教育部)推出了全国计算机等级考试,这是一种专门评价应试人员对计算机软硬件实际掌握能力的考试。它不限制报考人员的学历和年龄,从而为培养各行业计算机应用人才开辟了一条广阔的道路。

1994 年是推出全国计算机等级考试的第一年,当年参加考试的有 1 万余人,2017 年报考人数已达 620 万人。截至 2017 年年底,全国计算机等级考试共开考 50 次,考生人数累计达 7 665 万人,有 2 885 万人获得了各级计算机等级证书。

事实说明,鼓励社会各阶层人士通过各种途径掌握计算机应用技术,并通过等级考试对他们的能力予以科学、公正、权威性的认证,是一种比较好的、有效的计算机应用人才培养途径,符合我国的具体国情。等级考试同时也为用人单位录用和考核人员提供了一种测评手段。从有关公司对等级考试所作的社会抽样调查结果看,不论是管理人员还是应试人员,对该项考试的内容和



形式都给予了充分肯定。

计算机技术日新月异。全国计算机等级考试大纲顺应技术发展和社会需求的变化,从2010年开始对新版考试大纲进行调研和修订,在考试体系、考试内容、考试形式等方面都做了较大调整,希望等级考试更能反映当前计算机技术的应用实际,使培养计算机应用人才的工作更健康地向前发展。

全国计算机等级考试取得了良好的效果,这有赖于各有关单位专家在等级考试的大纲编写、试题设计、阅卷评分及效果分析等多项工作中付出的大量心血和辛勤劳动,他们为这项工作的开展作出了重要的贡献。我们在此向他们表示衷心的感谢!

我们相信,在21世纪知识经济和加快发展信息产业的形势下,在教育部考试中心的精心组织领导下,在全国各有关专家的大力配合下,全国计算机等级考试一定会以“激励引导成才,科学评价用才,服务社会选材”为目标,服务考生和社会,为我国培养计算机应用专门人才的事业作出更大的贡献。

前 言

随着我国计算机应用的进一步普及和深入,人们已经达成了一个共识:计算机知识是当代人类文化的重要组成部分,计算机应用是当代人才不可缺少的素质。因此,许多单位把计算机知识和应用能力作为考核、录用工作人员的重要条件,许多人也在努力证实自身在这方面的实力。人们都在寻求一个统一、客观、公正的衡量标准,各种考试也就应运而生,其中最受欢迎和信赖的是教育部考试中心组织的“全国计算机等级考试”。此考试中,不仅涉及计算机有关方面的基本知识,而且考核考生实际操作的能力。这一考试自 1994 年举办以来,应试人数逐年递增。为适应考试的需要,我们根据教育部考试中心制定的《全国计算机等级考试二级 C 语言程序设计考试大纲(2018 年版)》编写了这本教程。本书紧扣考试大纲,内容取舍得当,是一本系统的应试教材。

本书内容共分 16 章,包括:程序设计基本概念,C 语言程序设计的初步知识,顺序结构,选择结构,循环结构,字符型数据,函数,地址和指针,数组,字符串,对函数的进一步讨论,C 语言中用户标识符的作用域和存储类,编译预处理和动态存储分配,结构体、共用体和用户定义类型,位运算,文件等。本书的编写力求在体系结构上安排合理、重点突出、难点分散、便于读者由浅入深逐步掌握,在语言叙述上注重概念清晰、逻辑性强、通俗易懂、便于自学。本书所有例题涵盖了初学程序设计者应掌握的基本算法,并在 VC 的环境下运行通过。书中各章后均配有习题及参考答案,以便于读者自我检查,同时给出了若干编程题并提示了主要算法,以便读者上机练习。本书配套的电子教案可从如下地址下载,<https://pan.baidu.com/s/1geLtyMJ>(注意,网址中的“1”是数字而不是字母)。

本书由教育部考试中心组织编写并审定。第 1~16 章由田淑清、周海燕、孙甲松、叶海建合作编写。在本书的编写和出版过程中,教育部考试中心和高等教育出版社给予了大力支持,在此一并表示衷心的感谢。

由于我们水平有限,书中难免会有许多不足之处。我们真诚希望得到广大读者的批评指正。

编者

目 录

第 1 章 程序设计基本概念	1	运算符	20
1.1 程序和程序设计	1	2.7.1 自加运算符“++”和自减运算 符“--”	20
1.1.1 C 程序	1	2.7.2 逗号运算符和逗号表达式	21
1.1.2 程序设计	2	习题	21
1.2 算法	2	第 3 章 顺序结构	23
1.3 结构化程序设计和模块化结构	3	3.1 赋值语句	23
1.3.1 结构化程序	3	3.2 数据输出	24
1.3.2 模块化结构	6	3.2.1 printf 函数的一般调用形式	24
习题	7	3.2.2 printf 函数中常用的格式说明	25
第 2 章 C 程序设计的初步知识	8	3.2.3 使用 printf 函数时的注意事项	28
2.1 简单 C 语言程序的构成和格式	9	3.3 数据输入	29
2.2 标识符、常量和变量	10	3.3.1 scanf 函数的一般调用形式	29
2.2.1 标识符	10	3.3.2 scanf 函数中常用的格式说明	29
2.2.2 常量	11	3.3.3 通过 scanf 函数从键盘输入数据	30
2.2.3 符号常量	11	3.4 复合语句和空语句	32
2.2.4 变量	12	3.4.1 复合语句	32
2.3 整型数据	12	3.4.2 空语句	32
2.3.1 整型常量	12	3.5 程序举例	32
2.3.2 整型变量	13	习题	34
2.3.3 整型数据的分类	13	第 4 章 选择结构	37
2.3.4 整数在内存中的存储形式	14	4.1 关系运算和逻辑运算	38
2.4 实型数据	15	4.1.1 C 语言的逻辑值	38
2.4.1 实型常量	15	4.1.2 关系运算符和关系表达式	38
2.4.2 实型变量	15	4.1.3 逻辑运算符和逻辑表达式	39
2.5 算术表达式	16	4.2 if 语句和用 if 语句构成的选择 结构	40
2.5.1 基本的算术运算符	16	4.2.1 if 语句	40
2.5.2 运算符的优先级、结合性和算术 表达式	16	4.2.2 嵌套的 if 语句	43
2.5.3 强制类型转换表达式	17	4.3 条件表达式构成的选择结构	45
2.6 赋值表达式	18	4.4 switch 语句以及用 switch 语句 和 break 语句构成的选择结构	46
2.6.1 赋值运算符和赋值表达式	18	4.4.1 switch 语句	46
2.6.2 复合赋值表达式	19	4.4.2 switch 语句的执行过程	46
2.6.3 赋值运算中的类型转换	19		
2.7 自加、自减运算符和逗号			



4.4.3 在 switch 语句体中使用 break 语句	47	6.4 程序举例	76
4.5 语句标号和 goto 语句	48	习题	78
4.5.1 语句标号	48	第 7 章 函数	81
4.5.2 goto 语句	48	7.1 库函数	82
习题	49	7.2 函数的定义和返回值	83
第 5 章 循环结构	53	7.2.1 函数定义的语法	83
5.1 while 语句和用 while 语句 构成的循环结构	53	7.2.2 函数的返回值	84
5.1.1 while 循环的一般形式	53	7.3 函数的调用	84
5.1.2 while 循环的执行过程	54	7.3.1 函数的两种调用方式	84
5.2 do-while 语句和用 do-while 语句构成的循环结构	57	7.3.2 函数调用时的语法要求	85
5.2.1 do-while 语句构成的循环结构	57	7.4 函数的说明	85
5.2.2 do-while 循环的执行过程	57	7.4.1 函数说明的形式	85
5.3 for 语句和用 for 语句构成的 循环结构	58	7.4.2 函数说明的位置	86
5.3.1 for 语句构成的循环结构	58	7.5 调用函数和被调用函数之间的 数据传递	86
5.3.2 for 循环的执行过程	59	7.6 程序举例	88
5.3.3 有关 for 语句的说明	59	习题	93
5.4 循环结构的嵌套	60	第 8 章 地址和指针	97
5.5 break 和 continue 语句在循环 体中的作用	63	8.1 变量的地址和指针	97
5.5.1 break 语句	63	8.2 指针变量的定义和指针变量的 基类型	99
5.5.2 continue 语句	63	8.3 给指针变量赋值	100
5.6 程序举例	64	8.3.1 给指针变量赋地址值	100
习题	66	8.3.2 给指针变量赋“空”值	101
第 6 章 字符型数据	71	8.4 对指针变量的操作	101
6.1 字符型常量	72	8.4.1 通过指针来引用一个存储单元	101
6.1.1 字符常量	72	8.4.2 移动指针	103
6.1.2 转义字符常量	72	8.4.3 指针比较	104
6.1.3 字符串常量	73	8.5 函数之间地址值的传递	104
6.1.4 可对字符型量进行的运算	73	8.5.1 形参为指针变量时实参和形参 之间的数据传递	104
6.2 字符变量	73	8.5.2 通过传送地址值在被调用函数 中直接改变调用函数中的变量 的值	105
6.3 字符的输入和输出	74	8.5.3 函数返回地址值	106
6.3.1 调用 printf 和 scanf 函数输出和 输入字符	74	习题	107
6.3.2 调用 putchar 和 getchar 函数输出 和输入字符	75	第 9 章 数组	111
		9.1 一维数组的定义和一维数组 元素的引用	112



9.1.1 一维数组的定义	112	形参之间的数据传递	135
9.1.2 一维数组元素的引用	113	9.7.2 指针数组作为实参时实参和形参	
9.1.3 一维数组的初始化	113	之间的数据传递	136
9.1.4 通过赋初值定义数组的大小	114	9.8 二维数组程序举例	136
9.1.5 一维数组的定义和数组元素引用		习题	139
举例	114	第 10 章 字符串	147
9.2 一维数组和指针	115	10.1 用一维字符数组存放字符串	148
9.2.1 一维数组和数组元素的地址	115	10.1.1 通过赋初值的方式给一维字符	
9.2.2 通过数组的首地址引用数组		数组赋字符串	149
元素	116	10.1.2 在 C 程序执行过程中给一维字符	
9.2.3 通过指针引用一维数组元素	116	数组赋字符串	150
9.2.4 用带下标的指针变量引用一维		10.2 使指针指向一个字符串	151
数组元素	117	10.2.1 通过赋初值的方式使指针指向	
9.3 函数之间对一维数组和数组		一个字符串	151
元素的引用	117	10.2.2 通过赋值运算使指针指向一个	
9.3.1 数组元素作实参	117	字符串	151
9.3.2 数组名作实参	117	10.2.3 用字符数组作为字符串和用指针	
9.3.3 数组元素地址作为实参	119	指向的字符串之间的区别	151
9.3.4 函数的指针形参和函数体中		10.3 字符串的输入和输出	152
数组的区别	120	10.3.1 输入和输出字符串时的必要	
9.4 一维数组应用举例	121	条件	152
9.5 二维数组的定义和二维数组		10.3.2 用格式说明符 %s 进行整串	
元素的引用	129	输入和输出	152
9.5.1 二维数组的定义	129	10.3.3 调用 gets、puts 函数在终端	
9.5.2 二维数组元素的引用	130	输入或输出一行字符串	153
9.5.3 二维数组的初始化	130	10.4 字符串数组	153
9.5.4 通过赋初值定义二维数组的		10.5 用于字符串处理的函数	155
大小	131	10.6 程序举例	155
9.5.5 二维数组的定义和数组元素		习题	159
引用举例	131	第 11 章 对函数的进一步讨论	164
9.6 二维数组和指针	132	11.1 传给 main 函数的参数	164
9.6.1 二维数组和数组元素的地址	132	11.2 通过实参向函数传递函数名	
9.6.2 通过地址引用二维数组元素	133	或指向函数的指针变量	166
9.6.3 通过建立一个指针数组引用		11.3 函数的递归调用	167
二维数组元素	134	习题	170
9.6.4 通过建立一个行指针引用二维		第 12 章 C 语言中用户标识符的作用	
数组元素	135	域和存储类	174
9.7 二维数组名和指针数组作为		12.1 局部变量、全局变量和存储	
实参	135	分类	174
9.7.1 二维数组名作为实参时实参和			



12.1.1 用户标识符的作用域	174	14.2.5 函数之间结构体变量的数据 传递	200
12.1.2 局部变量、全局变量和存储 分类	175	14.2.6 利用结构体变量构成链表	204
12.2 局部变量及其作用域和 生存期	175	14.3 共用体	210
12.2.1 auto 变量	175	14.3.1 共用体类型的说明和变量定义 ..	210
12.2.2 register 变量	176	14.3.2 共用体变量的引用	211
12.2.3 静态存储类的局部变量	177	习题	212
12.3 全局变量及其作用域和 生存期	177	第 15 章 位运算	217
12.3.1 全局变量的作用域和生存期	177	15.1 位运算符	217
12.3.2 在同一编译单位内用 extern 说明符扩展全局变量的 作用域	179	15.2 位运算符的运算功能	218
12.3.3 在不同编译单位内用 extern 说明符扩展全局变量的 作用域	179	习题	221
12.3.4 静态全局变量	180	第 16 章 文件	222
12.4 函数的存储分类	180	16.1 C 语言文件的概念	222
12.4.1 用 extern 说明函数	180	16.2 文件指针	224
12.4.2 用 static 说明函数	181	16.3 打开文件	224
习题	181	16.4 关闭文件	225
第 13 章 编译预处理和动态存储分配 ..	184	16.5 调用 getc(fgetc) 和 putc(fputc) 函数进行输入和输出	226
13.1 编译预处理	184	16.6 判断文件结束函数 feof	228
13.1.1 宏替换	185	16.7 fscanf 函数和 fprintf 函数	228
13.1.2 文件包含	187	16.8 fgets 函数和 fputs 函数	229
13.2 动态存储分配	187	16.9 fread 函数和 fwrite 函数	230
13.2.1 malloc 函数和 free 函数	188	16.10 文件定位函数	231
13.2.2 calloc 函数	189	16.10.1 fseek 函数	231
习题	189	16.10.2 ftell 函数	232
第 14 章 结构体、共用体和用户定义 类型	192	16.10.3 rewind 函数	232
14.1 用 typedef 说明一种新 类型名	193	习题	232
14.2 结构体类型	194	附录 1 考试指导	235
14.2.1 结构体类型的说明	194	附录 2 C 语言的关键字	244
14.2.2 结构体类型的变量、数组和指针 变量的定义	195	附录 3 双目算术运算中两边运算量 类型转换规律	245
14.2.3 给结构体变量、数组赋初值	197	附录 4 运算符的优先级和结合性	246
14.2.4 引用结构体变量中的数据	198	附录 5 常用字符与 ASCII 代码 对照表	248
		附录 6 库函数	249
		附录 7 全国计算机等级考试二级 C 语言程序设计考试大纲 (2018 年版)	254



附录 8 全国计算机等级考试二级 C 语言程序设计样题及

参考答案	257
附录 9 习题参考答案	270



学习目标与要求

- 一、了解程序设计的基本含义
- 二、了解“算法”的基本特点,学习“算法”的重要性
- 三、结构化程序设计
 1. 了解结构化程序的3种基本结构。
 2. 掌握如何用一般流程图和 N-S 流程图表示三种基本结构。

1.1 程序和程序设计

1.1.1 C 程序

计算机现已广泛应用于社会生活的各个领域,成为大众化的现代工具。但是,不熟悉计算机的人仍然把它想象得十分神秘。其实,计算机不过是一种具有内部存储能力、由程序自动控制的电子设备。人们将需要计算机做的工作写成一定形式的指令,并把它们存储在计算机内部的存储器中,当人们给出命令之后,它就按指令顺序自动进行操作。人们把这种可以连续执行的一条条指令的集合称为“程序”。可以说,程序就是人与机器“对话”的语言,也就是我们常说的“程序设计语言”。

目前,在社会上使用的程序设计语言有上百种,它们中的大多数被称为计算机的“高级语言”,如 Visual Basic、C++、Java 以及本书将要介绍的 C 语言等。这些语言都是用接近人们习惯的自然语言和数学语言作为表达形式,使人们学习和操作起来感到十分方便。

但是,对于计算机本身来说,它并不能直接识别由高级语言编写的程序,它只能接受和处理由 0 和 1 的代码构成的二进制指令或数据。由于这种形式的指令是面向机器的,因此也被称为“机器语言”。

我们把由高级语言编写的程序称为“源程序”,把由二进制代码表示的程序称为“目标程序”。为了把源程序转换成机器能接受的目标程序,软件工作者编制了一系列软件,通过这些软



件可以把用户按规定语法写出的语句一一翻译成二进制的机器指令。这种具有翻译功能的软件称为“编译程序”,每种高级语言都有与它对应的编译程序。例如,C语言编译程序就是这样的一种软件,其功能如图1.1所示。

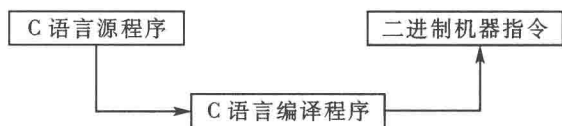


图 1.1 C语言编译程序功能示意图

我们所写的每条C语句,经过编译(Compile)最终都将转换成二进制的机器指令。由C语言构成的指令序列称为C源程序;按C语言的语法编写C程序的过程,称为C语言的代码编写。

C源程序经过C编译程序编译之后生成一个后缀为 .OBJ 的二进制文件(称为目标文件),然后由称为“连接程序”(Link)的软件,把此 .OBJ 文件与C语言提供的各种库函数连接起来生成一个后缀为 .EXE 的可执行文件。在操作系统环境下,只需点击或输入此文件的名字(而不必输入后缀 .EXE),该可执行文件就可运行。

1.1.2 程序设计

简单的程序设计一般包含以下几个部分。

(1) 确定数据结构。根据任务书提出的要求、指定的输入数据和输出结果,确定存放数据的数据结构。

(2) 确定算法。针对存放数据的数据结构来确定解决问题、完成任务的步骤。有关算法的概念将在下一节中介绍。

(3) 编码。根据确定的数据结构和算法,使用选定的计算机语言编写程序代码,输入到计算机并保存在磁盘上,简称编程。

(4) 在计算机上调试程序。消除由于疏忽而引起的语法错误或逻辑错误;用各种可能的输入数据对程序进行测试,使之对各种合理的数据都能得到正确的结果,对不合理的数据能进行适当的处理。

(5) 整理并写出文档资料。

1.2 算法

学习计算机程序设计语言的目的,是要用语言作为工具,设计出可供计算机运行的程序。

在拿到一个需要求解的问题之后,怎样才能编写出程序呢?除了选定合理的数据结构外,一般来说,十分关键的一步是设计算法,有了一个好的算法,就可以用任何一种计算机高级语言把算法转换为程序(编写程序)。

算法是指为解决某个特定问题而采取的确定且有限的步骤。一个算法应当具有以下五个特性:

(1) 有穷性。一个算法包含的操作步骤应该是有限的。也就是说,在执行若干个操作步骤之后,算法将结束,而且每一步都在合理的时间内完成。

(2) 确定性。算法中每一条指令必须有确切的含义,不能有二义性,对于相同的输入必能得出相同的执行结果。

(3) 可行性。算法中指定的操作,都可以通过已经验证过可以实现的基本运算执行有限次后实现。

(4) 有零个或多个输入。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。

(5) 有一个或多个输出。算法的目的是为了求“解”,这些“解”只有通过输出才能得到。

算法可以用各种描述方法来进行描述,最常用的是伪代码和流程图。

伪代码是一种近似于高级语言但又不受语法约束的一种语言描述方式,这在英语国家中使用起来更为方便。

流程图也是描述算法的很好的工具,一般的流程图由图 1.2 中所示的几种基本图形组成。



图 1.2 一般的流程图所用的几种基本图形

由这些基本图形中的框和流程线组成的流程图来表示算法,形象直观,简单方便。但是,这种流程图对于流程线的走向没有任何限制,可以任意转向,在描述复杂的算法时所占篇幅较多,费时费力且不易阅读。

随着结构化程序设计方法的出现,1973 年美国学者 I. Nassi 和 B. Shneiderman 提出了一种新的流程图形式,这种流程图完全去掉了流程线,算法的每一步都用一个矩形框来描述,把一个个矩形框按执行的次序连接起来就是一个完整的算法描述。这种流程图用两位学者名字的第一个英文字母命名,称为 N-S 流程图。在下一节中将结合结构化程序设计中的三种基本结构来介绍这种流程图的基本结构。

由于本书中的例题比较简单,因此大都采用文字来进行描述,但要求读者能读懂这两种流程图。

1.3 结构化程序设计和模块化结构

1.3.1 结构化程序

结构化程序由三种基本结构组成。

(1) 顺序结构。在本书第 3 章中将要介绍的如赋值语句、输入、输出语句都可构成顺序结



构。当执行由这些语句构成的程序时,将按这些语句在程序中的先后顺序逐条执行,没有分支,没有转移。顺序结构可用图 1.3 所示的流程图表示,其中(a)是一般的流程图,(b)是 N-S 流程图。

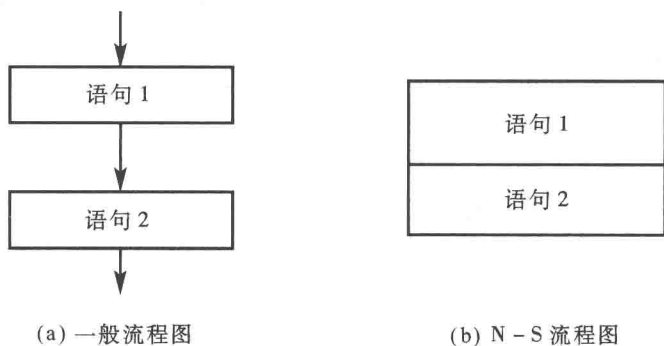


图 1.3 顺序结构流程图

(2) 选择结构。在本书第 4 章中将要介绍的 if 语句、switch 语句都可构成选择结构。当执行到这些语句时,将根据不同的条件去执行不同分支中的语句。选择结构可用图 1.4 所示的流程图表示,其中(a)是一般的流程图,(b)是 N-S 流程图。

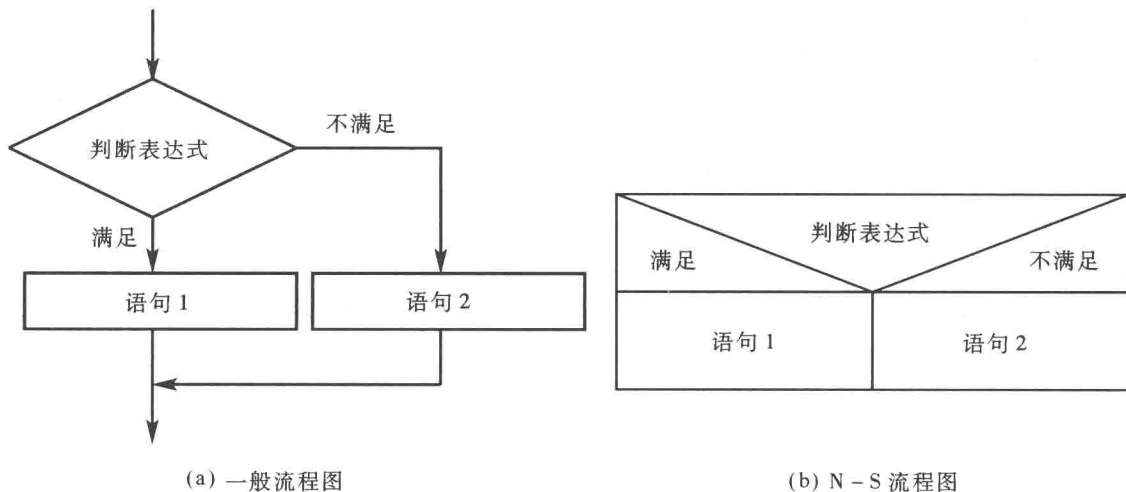
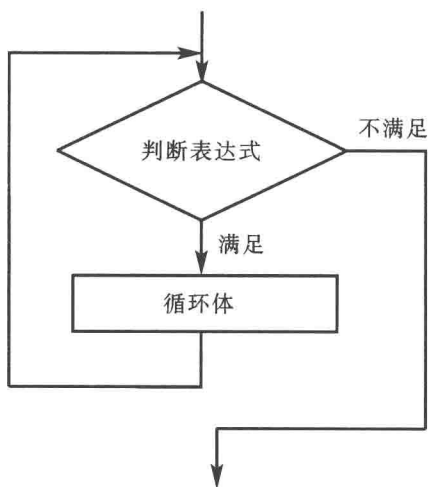


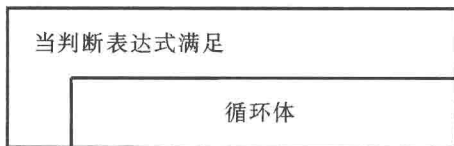
图 1.4 选择结构流程图

(3) 循环结构。在本书第 5 章中将介绍不同形式的循环结构。它们将根据各自的条件,使同一组语句重复执行多次或一次也不执行。循环结构的流程图如图 1.5 和图 1.6 所示,每个图中(a)是一般的流程图,(b)是 N-S 流程图。图 1.5 是当型循环流程图。当型循环的特点是:当指定的条件满足(成立)时,就执行循环体,否则就不执行。图 1.6 是直到型循环流程图。直到

型循环的特点是:执行循环体直到指定的条件满足(成立)时就不再执行循环体。

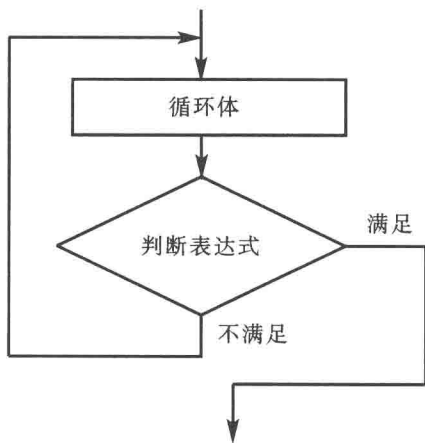


(a) 一般流程图

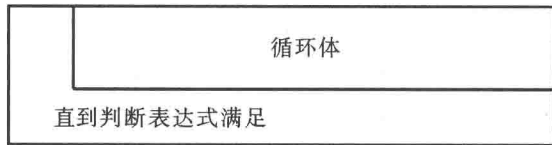


(b) N-S 流程图

图 1.5 当型循环流程图



(a) 一般流程图



(b) N-S 流程图

图 1.6 直到型循环流程图

已经证明,由三种基本结构组成的算法可以解决任何复杂的问题。由三种基本结构所构成的算法称为结构化算法;由三种基本结构所构成的程序称为结构化程序。

例 1.1 先后输入若干个整数,要求打印出其中最大的数,当输入的数小于 0 时结束。用 N-S 流程图表示算法。

解题的思路是:先输入一个数,在没有其他数参加比较之前,它显然是当前最大的数,把它放到变量 max 中。让 max 始终存放当前已比较过的数中的最大值。然后输入第二个数,并与 max



比较,如果第二个数大于 max,则用第二个数取代 max 中原来的值。如此先后输入和比较,每次比较后都将值大者放在 max 中,直到输入的数小于 0 时结束。最后 max 中的值就是所有输入数中的最大值。

根据此思路,画出 N-S 流程图(见图 1.7)。变量 x 用来控制循环次数,当 $x > 0$ 时,执行循环体;在循环体内进行两个数的比较和输入 x 值。从图 1.7 可见,在循环体的矩形框内包含一个选择结构。

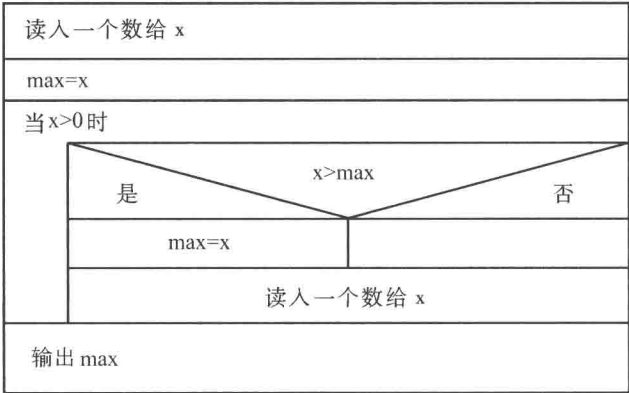


图 1.7 例 1.1 程序的 N-S 流程图

1.3.2 模块化结构

当计算机在处理较复杂的任务时,所编写的程序经常由上万条语句组成,需要由许多人来共同完成。这时常常把这个复杂的任务分解为若干个子任务,每个子任务又分成很多个小子任务,每个小子任务只完成一项简单的功能。在程序设计时,用一个个小模块来实现这些功能,每个程序设计人员分别完成一个或多个小模块。我们称这样的程序设计方法为“模块化”的方法,由一个个功能模块构成的程序结构为模块化结构。

由于把一个大程序分解成若干相对独立的子程序,每个子程序的代码一般不超过一页纸,因此对程序设计人员来说,编写程序代码变得不再困难。这时只需对程序之间的数据传递做出统一规范,同一软件可由一组人员同时进行编写,分别进行调试,这就大大提高了程序编制的效率。

软件编制人员在进行程序设计的时候,首先应当集中考虑主程序中的算法,写出主程序后再动手逐步完成子程序的调用。对于这些子程序也可用调试主程序的同样方法逐步完成其下一层子程序的调用。这就是自顶向下、逐步细化、模块化的程序设计方法。

C 语言是一种结构化程序设计语言。它提供了三种基本结构的语句;提供了定义“函数”的功能,在 C 语言中没有子程序的概念,它提供的函数可以完成子程序的所有功能;C 语言允许对函数单独进行编译,从而可以实现模块化。另外,C 语言还提供了丰富的数据类型。这些都为结构化程序设计提供了有力的工具。