



计算机基础教育系列教材

C 语言程序设计教程

吴国凤 周恒忠 冯崇岭 石 竹 孙家启 编著

安徽大学出版社

计算机基础教育系列教材

C 语言程序设计教程

吴国凤 周恒忠 冯崇岭 石 竹 孙家启 编著



安徽大学出版社

图书在版编目(CIP)数据

C 语言程序设计教程/吴国凤、周恒忠、冯崇岭、石竹、
孙家启等编著. —合肥:安徽大学出版社,1999.3
计算机基础教育系列教材
ISBN 7-81052-227-2

I. C… II. ①吴…②周…③冯…④石…⑤孙…
III. C 语言-程序设计-电子计算机-教材 IV. TP312

中国版本图书馆 CIP 数据核字(1999)第 01372 号

C 语言程序设计教程

吴国凤 周恒忠 冯崇岭 石 竹 孙家启 编著

出版发行	安徽大学出版社 (合肥市肥西路 3 号 邮编 230039)	印刷	中国科技大学印刷厂
联系电话	编辑室 0551-5106428 发行部 0551-5107784	开 本	787×1092 1/16
责任编辑	李 虹	印 张	14.375
封面设计	张 犇	字 数	349 千
经 销	新华书店	版 次	1999 年 3 月第 1 版
		印 次	2001 年 11 月第 3 次印刷
ISBN 7-81052-227-2/TP·20		定价 17.80 元	

如有影响阅读的印装质量问题,请与出版社发行部联系调换

内容简介

本书是计算机基础教育系列教材之一,根据教育部计算机基础课程教学指导委员会颁布的“C 语言程序设计”教学要求,全面地、系统地叙述 C 语言的语法知识及其程序设计方法和技能。

本书兼顾了全国高等学校(安徽考点)计算机等级考试需要,其内容覆盖了省教育厅制定的二级 C 语言程序设计教学(考试)要求,所以它又是一本考试指导书——全国高等学校计算机等级(水平)考试系列教材之一。本书另配有《C 语言程序设计题解与实验》。

全书共分 12 章。内容丰富、系统性强、深入浅出,各章均有要点及例题分析,还附有针对性习题,便于读者学习和自测,对备考极具参考价值。

本书特别适用于普通高校本、专科非计算机专业的教材或成教、夜大、函大计算机专业的教材,也可供参加全国高等学校计算机等级(水平)考试考点学校使用,还可供广大计算机自学者、工程技术人员参考。

编委会名单

主 任:孙家启

委 员:	王忠仁	王永国	方潜生	尹荣章
	石 竹	冯年荣	冯崇岭	孙家启
	孙道德	仲 红	江 涛	朱学勤
	齐学梅	吴国凤	李 雪	何 明
	张伟林	张国平	陈桂林	陈高潮
	郑尚志	胡宏智	周鸣争	周恒忠
	姚合生	赵林玲	聂会星	徐奇观
	徐精明	黄海生	程承士	谢荣传
	蔡之让	潘 瑜	(以姓氏笔划写序)	

秘书长:王忠仁 聂会星

编 写 说 明

为了支持计算机基础教育改革与建设,促进计算机基础课程教学与水平考试向纵深发展,我们按照计算机文化基础教育、技术基础教育和应用基础教育三个层次,组织编写了计算机基础教育系列教材。这套教材囊括了计算机文化基础、高级语言(QBasic, Visual Basic, C, Visual C⁺⁺, PASCAL, FORTRAN77, FORTRAN90, FoxBASE⁺, FoxPro 2.5b For Windows, Visual FoxPro 等)程序设计、软件技术基础、微型计算机原理及应用、计算机网络等方面内容,涵盖全国高校计算机水平考试的一、二、四级(全国等级考试的一、二、三级),因而具有广泛的适应性。这套教材所具有的突出特点是:紧扣计算机基础教育大纲(即计算机水平考试大纲),兼具普通教材与考试辅导材料的双重功能;立意创新,内容简练,大量针对性极强的习题和典型例题分析是其他教材所少见;编写人员都是教学、科研第一线有着丰富教学与实践经验的教师,他们深谙相关知识点的张弛取舍。我们还聘请了三位知名专家担任高级顾问,以确保本系列教材的编写质量。

本系列教材的先期版本现已问世,第一辑各册已于1999年底全部出齐。由于计算机技术的发展比人们想象的还要快,所以本系列教材又增加了不少新内容,我们今后还将不断调整、更新教材内容、平台和版本,使之与当时发展相适应,以便教材以更新更好的面目呈现在读者面前。

本系列教材编写目的明确,它特别适合于作为普通高校非计算机专业的本、专科教学用教材或成教、夜大、函大计算机专业的教材,也可供各地计算机水平考试考点使用,还可供广大计算机自学者、工程技术人员参考。

编写委员会
2000年5月

前 言

C 语言是近年来国内外得到最迅速推广使用的一种现代编译型程序设计语言,它兼顾了多种高级语言的特点,具备汇编语言的功能。C 语言程序处理功能强、运算速度快、目标效率高,具有完善的模块程序结构,可移植性好,而且可以直接实现对系统硬件及外部设备接口的控制,具有较强的系统处理能力。在当今世界技术先进国家中,使用 C 语言进行程序设计已成为软件开发的一个主流。1983 年在 C 语言的基础上推出的面向对象的程序设计语言 C⁺⁺,在 90 年代得到迅猛发展。C⁺⁺ 语言今后将成为最流行的一种计算机主流语言。掌握好 C/C⁺⁺ 语言已成为当今软件工作者的必备条件之一。

为了在我国高等学校更快地推广和普及 C 语言,编者根据国家教育部全国高等学校计算机基础课程教学指导委员会制订的 C 语言程序设计教学要求,兼顾全国高等学校(安徽考点)计算机等级(二级 C 语言程序设计)教学(考试)大纲编写此教材的。由于我国目前高等学校广泛使用的 PC 系列各档次(486,586)微型机上的 C 编译系统 TurboC 2.0 为实现版本,全面地、系统地讨论了 C 语言语法知识以及程序设计方法和技巧。书中各章均有要点及例题分析,还附有针对性的习题便于读者学习和自测,对参加全国高校计算机等级(二级 C 语言程序设计)考试极具参考价值。根据广大读者要求,本书第 3 次印刷时,将第 1 章内容作较大的调整,并改名为 C 语言概述,在第四章中增加了用指针处理链表一节,以供学习应用,书中部分章节习题稍作调整并另配有《C 语言程序设计题解与实验》。

本书由吴国凤、周恒忠、冯崇岭、石竹、孙家启合作编写。全书由孙家启统稿并修改。在成书过程中,得到了省高校同行专家们的大力支持,在此表示感谢。

由于编写时间仓促,加之水平有限,难免有疏漏、错误之处,恳请读者批评指正。

编者

2001 年 10 月

目 次

第 1 章 C 语言概述	(1)
1.1 C 语言的来由	(1)
1.2 C 语言的特点	(1)
1.3 C 语言的基本程序结构	(2)
1.4 算法	(4)
1.5 C 程序的上机步骤	(9)
1.6 本章要点及例题分析	(12)
习题 1	(14)
第 2 章 数据类型和运算	(15)
2.1 C 语言的数据类型	(15)
2.2 变量的初值和类型转换	(20)
2.3 基本运算符和表达式	(22)
2.4 本章要点及例题分析	(26)
习题 2	(28)
第 3 章 输入和输出	(31)
3.1 printf 函数	(31)
3.2 scanf 函数	(33)
3.3 scanf, getchar, printf 和 putchar 的比较	(36)
3.4 本章要点及例题分析	(36)
习题 3	(39)
第 4 章 语句和流程控制	(42)
4.1 C 语言语句	(42)
4.2 分支结构程序	(43)
4.3 循环结构程序	(48)
4.4 转移语句	(51)
4.5 程序举例	(52)
4.6 本章要点及例题分析	(54)
习题 4	(58)
第 5 章 数组	(64)
5.1 一维数组	(64)
5.2 二维数组	(67)
5.3 字符数组	(70)
5.4 程序举例	(75)
5.5 本章要点及例题分析	(76)
习题 5	(81)

第6章 函数	(85)
6.1 概述	(85)
6.2 函数定义	(86)
6.3 函数调用	(87)
6.4 函数的参数和函数的值	(88)
6.5 函数说明	(89)
6.6 数组作为函数参数	(90)
6.7 函数的嵌套调用	(93)
6.8 函数的递归调用	(95)
6.9 内部函数和外部函数	(98)
6.10 本章要点及例题分析	(99)
习题6	(104)
第7章 变量作用域和存储类型	(109)
7.1 变量作用域	(109)
7.2 变量的存储类型	(111)
7.3 本章要点及例题分析	(115)
习题7	(118)
第8章 编译预处理	(122)
8.1 概述	(122)
8.2 宏定义	(122)
8.3 文件包含	(125)
8.4 条件编译	(126)
8.5 本章要点及例题分析	(128)
习题8	(130)
第9章 指针	(133)
9.1 指针的基本概念	(133)
9.2 指针变量的类型说明	(133)
9.3 指针变量的引用	(134)
9.4 指针和函数参数	(135)
9.5 数组指针变量	(136)
9.6 数组名和数组指针变量作函数参数	(138)
9.7 指向多维数组的指针变量	(139)
9.8 字符串指针变量	(140)
9.9 使用字符串指针变量与字符数组	(142)
9.10 函数指针变量	(142)
9.11 指针型函数	(143)
9.12 指针数组	(144)
9.13 命令行参数— main 函数的参数	(146)
9.14 指向指针的指针变量	(147)

9.15 本章要点及例题分析·····	(147)
习题 9 ·····	(153)
第 10 章 结构、联合与枚举·····	(157)
10.1 结构·····	(157)
10.2 动态存储分配·····	(165)
10.3 用指针处理链表·····	(166)
10.4 联合·····	(178)
10.5 枚举·····	(181)
10.6 类型定义符 typedef ·····	(182)
10.7 本章要点及例题分析·····	(183)
习题 10 ·····	(185)
第 11 章 位运算 ·····	(189)
11.1 位运算符·····	(189)
11.2 位域·····	(190)
11.3 本章要点及例题分析·····	(192)
习题 11 ·····	(194)
第 12 章 文件 ·····	(197)
12.1 文件的基本概念·····	(197)
12.2 文件指针·····	(197)
12.3 文件的打开与关闭·····	(198)
12.4 文件的读写·····	(199)
12.5 文件的随机读写·····	(205)
12.6 文件的检测函数·····	(206)
12.7 C 库文件·····	(207)
12.8 本章要点及例题分析·····	(207)
习题 12 ·····	(209)
附录 A 常用字符与 ASCII 代码对照表 ·····	(212)
附录 B C 语言常用语法提要 ·····	(213)
附录 C C 库的常用函数 ·····	(216)
主要参考文献·····	(220)

第 1 章 C 语言概述

本章将简要介绍 C 语言来由;C 语言的特点;C 语言的基本程序结构;算法和 C 程序的上机步骤等,以便读者对 C 语言有一个总的印象,为以后各章学习打下基础。

1.1 C 语言的来由

C 语言是国际上广泛流行的计算机高级程序设计语言。它既是一个非常成功的系统描述语言,适合于编写系统软件(如操作系统、编译软件等),又是一个相当有效的通用程序设计语言,适合于编写各种应用软件(如图形软件、控制软件等)。

与其他高级语言相比,C 语言硬件控制能力和运算表达能力强,可移植性好,效率高(目标程序短,运行速度快)。因此其应用面越来越广,许多大型软件都用 C 语言编写。

C 语言是 1971 年由贝尔实验室的 D·M·Ritchie 在 B 语言的基础上设计的一种程序设计语言。1972 年投入使用。之后,C 语言成为 Unix 或 Xenix 操作系统的主要语言。是当今最为广泛的程序设计语言之一。

1.2 C 语言的特点

一种语言之所以能存在和发展,并具有生命力,总是有些不同于(若优于)其他语言的特点。C 语言有以下几个基本特点:

(1) C 语言简洁、紧凑,使用方便、灵活,C 语言一共只有 32 个保留字,9 种控制语句,程序书写形式自由,主要用小写字母表示,压缩了一切不必要的成分,相对其他语言源程序短,因此输入程序时工作量少。

(2) C 语言是处于汇编语言和高级语言之间的一种记述性程序设计语言,他允许直接访问地址,能进行位(Bit)运算,能实现汇编语言大部分功能,可以直接对硬件进行操作。

(3) C 语言是一种结构化程序设计语言,程序的逻辑结构可以用顺序、分支和循环三种基本结构组成。C 语言具有结构化控制语句(如 if~else, while, do~while, switch, for 等语句),十分便于采用由顶向下、逐步求精的结构化程序设计方法。C 语言程序的函数结构,十分利于把整体程序分割成若干相对独立的功能模块,并且为程序模块间的相互调用以及数据传递提供了便利。因此,用 C 语言编制的程序,具有容易理解、便于维护的优点。

(4) C 语言运算符丰富,它的运算符包含的范围很广泛,共有 34 种运算符。除一般高级语言使用的 +, -, *, / 四则运算及与(and)、或(or)、非(not)等逻辑运算功能外,还可以实现以二进制位(bit)为单位的位与(&)、位或(|)、位非(~)、位异或(^)以及移位(>>, <<)等位运算,并且具有如 a++, b-- 等单项运算和 +=, -=, *=, /= 等复合运算功能。

(5) C 语言的数据类型丰富,它的数据类型有:整型、实型、字符型、数组类型、指针类型、结构体类型、联合体类型和枚举类型等,能用来实现各种复杂的数据结构。因此,C 语言具

有较强的数据处理能力。

(6)C 语言程序中可以使用如 #define、#include 等编译预处理语句,能进行字符串或特定参数的宏定义,以及实现对外部文本文件的读取和合并。同时还具有 #if、#else 等条件编译预处理语句。这些功能的使用提高了软件开发的工作效率,并为程序的组织和编译提供了便利。

(7)C 语言程序可移植性好,C 语言程序本身并不依存于机器硬件系统,从而便于在硬件结构不同的机种间和各种操作系统实现程序的移植。

由于 C 语言具有上述众多特点,近年来迅速地得到普及和应用。许多大型的软件系统都用 C 语言编写,许多以前只能用汇编语言处理的问题现在可以改用 C 语言来处理。C 语言被称为“高级汇编语言”。

C 语言的优点很多,但和其他程序设计语言一样,它也有弱点,如运算符的优先级较多,有些还与常规约定不同,不便记忆;某些语法部分不易用形式化方法描述;各种 C 语言版本之间略有差别,缺乏统一的标准;C 语言不是强类型的语言,它强调灵活、高效的同时,在一定程度上牺牲了某些安全性,如类型检验太弱,转换比较随便等。因此,C 语言对程序设计人员提出了较高的要求,尤其在使用 C 语言的某些高级手段时更是如此。但是,C 语言的优点远远超过了它的弱点,这些优点使 C 语言具有强大的吸引力。实际经验表明,程序设计人员一旦接触了这种语言,并且有一定程序设计经验后,就会对它爱不释手。

1.3 C 语言的基本程序结构

任何一种计算机程序设计语言,都具有特定的语法规则、语义和一定的表现形式。程序的书写格式和程序的构成规则是程序语言表现形式的一个重要方面。按照规定的书写格式和构成规则书写程序,不仅可以使程序设计人员和使用程序的人容易理解,更重要的是把程序输入给计算机时,计算机能够充分认识,从而能够正确执行它。

先从一个最简单的程序看起,该程序的功能是在屏幕上显示:Hello!

程序如下:

```
main()  
{  
    printf("Hello!"\n);  
}
```

任何一个 C 语言源程序,都必须经过编译、连接后方可执行,不同种类的 C 语言,其编译、连接方式也不同。由于当前大部分的教学和考试都是以 Turbo C 2.0 为主,故本书以此作为主要考虑对象。要想运行该程序,按下 Ctrl + F9,系统对程序进行编译、连接,若程序没有什么错误,则在屏幕上输出 Hello!,然后回到程序编辑状态,想看程序运行结果,可按 Alt + F5 键。

现在,对程序本身作一解释。一个 C 语言程序,无论其大小,皆由一个或多个“函数”组成,而“函数”完成要做的操作。上例中,main()就是一函数。但 main 是一个特殊名,任何程序有且仅有一个 main,且程序总是从 main 的开头执行。main 的具体结构为:

```
main( )           /* 函数名 */
{                 /* 函数开始 */
    :             /* 函数体 */
}                 /* 函数结束 */
```

一个函数可调用另一个函数。函数间也可用参数来交换数据。函数名后的一对括号内括着一个参数表。例子中的 main 函数没有使用参数,故为一对空括号(不能省)。花括号 { } 用以包围构成函数的语句(即函数体)。例子中,函数体是由一条语句组成的。

```
printf("Hello! \n");
```

它是一个函数调用,它调用一个名叫 printf 的函数,带有参数“Hello! \n”。printf 是一个屏幕上输出的库函数。它打印输出作为其参数的字符串(由括号所括的字符序列)。

串中的序列“\n”是 C 语言中用作换行符的记号,当遇到它时,屏幕上的光标前进到下一行的最左边。因此,如漏掉了“\n”,会发现本语句输出结束时并未换行,若多次调用后,得到一个长的输出行。

例如:

```
printf("Hello,");
printf("This is a test.");
printf("\n");
其效果等于 printf("Hello, This is a test. \n");
```

它们都是在屏幕上输出:Hello, This is a test.

实际上,C 语言中,一个反斜线“\”后带一个字符,表示转义序列。转义序列是用来表示 ASCII 字符集内的控制代码如“回车”、“退格”、“制表”等不易获得或不可见的特殊字符。C 语言中定义的转义序列有:

\n 换行	\\ 反斜线
\t 制表	\' 单引号
\r 回车	\b 退格
\f 换页	\0 空(null)

要注意的是,虽然转义序列由反斜线“\”和另一个字符组成,但它却等效于一个单独字符,与其他单个字符性质相同。

转义序列也可和其他字符混合使用,例如:

```
printf("A \n BC \n AD \t \ EFG \n");
```

输出为:

```
A
BC
AD      EFG
```

最后说一下,C 语言对大小写是敏感的,即字母大小写是不等价的,在程序中一般情况下用小写。

1.4 算 法

1.4.1 算法概念

要计算机帮助我们解决问题,首先要编写出计算机程序。众所周知,计算机程序是许多指令的集合,每一条指令让计算机执行完成一个具体的操作,一个程序所规定的动作全部执行完成后,就能产生计算结果。因此,编写出正确的程序就成为让计算机帮助人们解决问题的关键。编制正确的程序必须具备两个基本条件:一是要掌握一门计算机高级语言的语法规则;二是要掌握解题的方法和步骤。

计算机语言只是一种工具。读者只学习语言的语法规则是不够的,最重要的是学会针对各种类型的问题,拟定出有效的解题方法和步骤即算法。算法的研究是计算机科学的核心,而算法概念本身也是计算机程序设计中最基本的概念之一。

正确的算法必须满足下列三个条件:

- ①每个逻辑块(模块)必须由可以实现的语句来完成;
- ②每个模块间的关系应该是惟一的(即块与块之间的关系一定要确定);
- ③算法要能终止(不能造成死循环)。

下列过程就不是一个正确的算法。计算过程:

第1步:令 n 等于 0;

第2步: n 加 1;

第3步:转向第2步。

如果你用计算机执行此过程,从理论上讲,计算机将永远执行下去(死循环)。

而下列过程就是一个正确的算法,它能结束:

第1步:设 n 等于 0;

第2步: n 加 1;

第3步:如果 n 小于 100,则转向第2步,否则停止。

这里不要把“计算方法”(Computational method)和“算法”(Algorithm)这两个词混淆。前者指的是求数值解的近似方法,后者是指解决问题的一步一步过程。

1.4.2 算法简例

【例 1.1】求 $1 \times 2 \times 3 \times \cdots \times 20$ 。

此例如果用原始的方法: $1 \times 2 \Rightarrow 2, 2 \times 3 \Rightarrow 6, 6 \times 4 \Rightarrow 24, \cdots$,一步一步去写,算法虽然是正确的,但太繁琐,它要写 19 个步骤才能完成,因而这种方法不可取。在实际使用中,一般对这种例题,采用如下一种通用的算法。

设置两个变量:一个变量代表被乘数,一个变量代表乘数。然后再把每次乘积的结果放在被乘数变量中,并用一种循环方式来表示算法。设 t 为被乘数, i 为乘数。算法。

① $1 \Rightarrow t$

② $2 \Rightarrow i$

③ $t \times i \Rightarrow t$

④ $i+1 \Rightarrow i$

⑤ 若 $i \leq 20$, 返回③; 否则, 结束。

在这个算法中③到⑤组成一个循环, 在实现算法中反复多次执行③, ④, ⑤步骤, 直到 i 值大于 20 为止, 算法结束, 最后一次变量 t 的值就是所求的结果。

如果题目改为 $1 \times 3 \times 5 \times 7 \times 9 \times 11 \times 13 \times 15 \times 17 \times 19 \times 21$ 。

算法只需稍作改动即可。算法。

① $1 \Rightarrow t$

② $3 \Rightarrow i$

③ $t \times i \Rightarrow t$

④ $i+2 \Rightarrow i$

⑤ 若 $i \leq 21$, 返回③; 否则, 结束。

可以看出这种方法表示的算法具有通用性、灵活性。

【例 1.2】 求两个正整数 m 和 n 的最大公因子。算法:

① 输入 m, n 的值;

② 用求余数%, 求 m 除以 n 的余数, 然后把余数赋给变量 r , 即 $m \% n \Rightarrow r$;

③ 判断 r 是否为 0;

④ 如果 $r \neq 0$, 则把 $n \Rightarrow m, r \Rightarrow n$, 返回②;

⑤ 若 $r = 0$, 则输出最大公因子, 结束。

此种算法表示的是求二个整数 m 和 n 最大因子的欧几里德算法。

【例 1.3】 某班有 32 个学生, 试将其中成绩最高的打印出来。算法:

① g_1 (表示第一个学生成绩) $\Rightarrow m$;

② $2 \Rightarrow i$;

③ $\left[\begin{array}{l} \text{当 } m > g_i \text{ (表示第 } i \text{ 个学生成绩) 时, 则执行④。} \\ \text{否则, } g_i \Rightarrow m \end{array} \right. ;$

④ $i+1 \Rightarrow i$;

⑤ $\left[\begin{array}{l} \text{当 } i \leq 32 \text{ 时, 返回③执行。} \\ \text{否则, 计算结束。} \end{array} \right.$

这种算法称之为选极值递推结构, 它的作用是将一批数中最大或最小的值挑选出来。上述算法是选最大值, 若将③改为:

$\left[\begin{array}{l} \text{当 } m < g_i \text{ 时, 则执行④。} \\ \text{否则, } g_i \Rightarrow m. \end{array} \right.$

则上述算法即为选最小值。

1.4.3 算法的特征

从以上所例举的试题中, 不难看出一般计算机的算法应具有以下五个特征。

1. 有穷性

一个算法必须在有限次执行后完成。

上面的例 1.2 表明, 当执行到第三轮 $r = 0$ 时, 求得 28 和 16 的最大公因子为 4 时算法

结束。显然它是满足了有穷性的条件。这是因为在第①步输入正整数 m 和 n 以后,接着按第②步执行,在第②步后, r 的值一定小于 n 。当 $r \neq 0$ 时,下一轮再执行第②步时, n 的值将减少。而正整数的递减序列必然导致最后的终止。当然,选择较大的 m 和 n 值可能增加第一步骤的执行次数。

2. 确定性

一个算法中的每一个步骤必须有明确的定义。计算机和自然语言不大相同,一切都要在程序中予以安排,不能有语意不明的地方。

这就是说,要执行的动作应该是非常明确的。它没有语义上解释的困难。诸如执行“ x 加 6 或者加 7 赋给变量 y ”这类的步骤是含糊不清的。因为它没有明确 x 应该加 6 或 7 中的哪一个数,在算法中类似这类的含糊不清的步骤是不允许的。在上面的实例中,第①步到第⑤步所表明的各个步骤其含义都是十分明确的。

3. 输入

算法总是要施加到运算对象上,是提供运算对象的初始情况,是算法的起点,所以一个算法应有 0 个或多个输入。

在例 1.2 中应有两个输入数据,它们是正整数 m 和 n 。

4. 输出

一个算法要有一个或多个输出。在例 1.2 中,它有一个输出,就是最后一轮在第③步中判别 $r=0$ 后输出的结果 n ,若无输出,人们是无法知道答案的。

5. 可行性

一个算法的可行性是指所有待实现的运算必须是相当基本的,至少在原则上人们可以用纸和笔做有限次操作即可完成的。

在例 1.2 中,第①步、第⑤步完成两个输入的整数和输出这两个整数的最大公因子的操作。第②步到第④步所示的一些步骤中用到了两个整数的除法运算,判别一个整数是否为零以及一个变量赋值给另一个变量的运算。这些运算都是一些基本的可行的运算。

算法实质上反映的是解决问题的思路。许多问题,只要仔细分析对象数据,就容易找到处理方法,对于定义明确的任务往往是这样。

1.4.4 怎样表示一种算法

为了表示一个算法。可以用不同的方法。常用的有:自然语言;伪代码(Pseudo code);N—S 结构化流程图(N 和 S 是两位美国学者的英文姓名的第一个字母);PAD 图(Problem Analysis Diagram 意为问题分析图);程序流程图。用计算机语言(包括低级语言和高级语言)写的程序也是算法的表示形式。下面介绍程序设计中常用的 3 种算法。

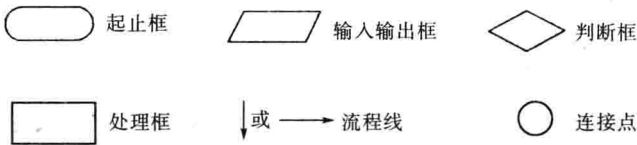


图 1-1

1. 用流程图表示算法

流程图是用一些图框来表示各种操作。用图形表示算法,直观形象,易于理解。美国国家标准化协会 ANSI(American National Standard Institute)规定了一些常用的流程图符号如图 1-1 所示,已为世界各国程序工作者普遍采用。

图 1-1 中菱形框的作用是对一个给定的条件进行判断,根据给定的条件是否成立来决定如何执行其后的操作。它有一个入口,两个出口。如图 1-2 所示。

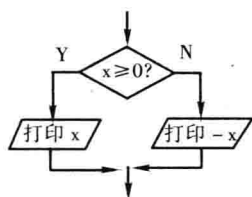


图 1-2

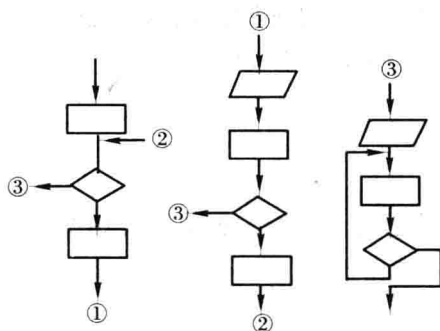


图 1-3

连接点(小圆圈)是用于将画在不同地方的流程线连接起来。如图 1-3 所示中有两个以○为标志的连接点(在连接点圈中写上“1”),它表示这两个点是互相连接在一起的。实际上它们是同一个点,只是画不下才分开来画。用连接点,可以避免流程线的交叉或过长,使流程图清晰。

【例 1.4】 将例 1.1 求 $20!$ 的算法用流程图表示,流程图如图 1-4 所示。

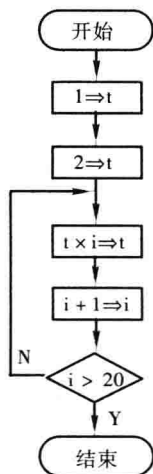


图 1-4

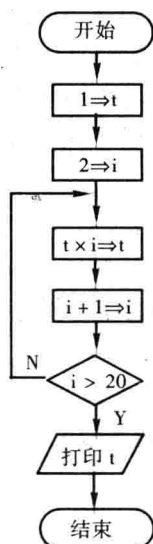


图 1-5

菱形框两侧的“Y”和“N”代表“是”(yes)和“否”(no)。如果需要将最后结果打印出来,可以在菱形框的下面再加一个输出框,如图 1-5 所示。

2. 用伪代码表示算法

流程图可表示一个算法,但在设计算法过程中使用不是很理想(尤其是当算法比较复杂、需要反复修改时)。为了设计算法时方便,常用一种称为伪代码(Pseudo code)的工具。

伪代码是用于自然语言和计算机语言之间的文字和符号来描述算法。它如同一篇文章,自上而下地写下来。每一行(或几行)表示一个基本操作。它不用图形符号,因此书写方便、格式紧凑,也比较易懂,便于向计算机语言算法(即程序)过渡。

例如,“打印 x 的绝对值”的算法可以用伪代码表示如下:

IF x is positive THEN

 print x

ELSE

 print $-x$

它像一个英语句子一样易懂,在国外用得比较普遍。也可以用汉字伪代码表示如下:

若 x 为正

 打印 x

否则

 打印 $-x$

也可以中英文混用表示如下:

IF x 为正

 print x

ELSE

 print $-x$

即计算机语言中具有 的语句关键字用英文表示,其他的可用汉字表示。总之,以便于书写和阅读为原则。用伪代码写算法并无固定的、严格的语法规则,只要把意思表达清楚,并且书写的格式要写成清晰易读的形式。

【例 1.5】 求 $20!$ 。用伪代码表示的算法如下:

开始

 置 t 的初值为 1

 置 i 的初值为 2

 当 $i \leq 20$, 执行下面操作:

 使 $t = t \times i$

 使 $i = i + 1$

 (循环体到此结束)

 打印 t 的值

结束

也可以写成以下形式:

BEGIN(算法开始)

$1 \Rightarrow t$

$2 \Rightarrow i$