



高等职业教育“十二五”规划教材

C语言程序设计

(理实一体化教程)

杜恒◎主编



赠电子课件



机械工业出版社
CHINA MACHINE PRESS

高等职业教育“十二五”规划教材

C 语言程序设计

（理实一体化教程）

主 编 杜 恒

副主 编 李 金 梁

参 编

李金梁 李 慧 方华丽

王 慧

主 审 李怀刚



机械工业出版社

C 语言是一种面向过程的高级编程语言,它是按照结构化的编程思想、模块化的程序设计方法来进行程序的编写和代码的组织。C 语言程序设计主要内容有:数据类型、运算符及表达式、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、函数、指针、编译预处理、结构体和共用体、位运算、文件操作等。C 语言功能强大,数据类型和运算符丰富、语言表达能力强,指针的使用又使 C 语言非常灵活,指针是 C 语言的精华。C 语言的特点使它既适合编写系统软件,又适合编写应用软件。

本书适合应用型本科、高职高专、中职等院校计算机、电子信息、机械、电气类专业学生学习 C 语言程序设计,也是计算机等级考试和编程爱好者的重要参考书。

图书在版编目(CIP)数据

C 语言程序设计:理实一体化教程/杜恒主编.—北京:机械工业出版社,2011.8
高等职业教育“十二五”规划教材
ISBN 978-7-111-34849-8

I. ①C… II. ①杜… III. ①C 语言—程序设计—高等职业教育—教材
IV. ①TP312

中国版本图书馆 CIP 数据核字(2011)第 146705 号

机械工业出版社(北京市百万庄大街 22 号 邮政编码 100037)

策划编辑:王玉鑫 责任编辑:王玉鑫

封面设计:王伟光 责任印制:杨 曦

北京圣夫亚美印刷有限公司印刷

2011 年 9 月第 1 版第 1 次印刷

184mm×260mm·20.5 印张·507 千字

0001—3000 册

标准书号:ISBN 978-7-111-34849-8

定价:37.00 元

凡购本书,如有缺页、倒页、脱页,由本社发行部调换

电话服务

网络服务

社服务中心:(010)88361066

门户网:<http://www.cmpbook.com>

销售一部:(010)68326294

教材网:<http://www.cmpedu.com>

销售二部:(010)88379649

封面无防伪标均为盗版

读者购书热线:(010)88379203

前 言

C 语言是当今世界上流传最广泛、使用最多的面向过程的高级程序设计语言之一。C 语言功能强大,语法灵活,数据类型和运算符丰富,语言表达能力强,用 C 语言编写的程序执行效率高、可移植性好。C 语言是一种高级语言,但是又具备低级语言的特点,因此 C 语言既适合编写系统软件,又适合编写应用软件,是不可多得的高级程序设计语言。

本书的编写按照理论与实际一体化教材的编写思路进行,注重理论联系实际,尤其是学生技能的培养、创新性思维的启发更能在本书中得到体现。本书第 1 章是 C 语言的总体概括,并介绍了 VC++ 6.0 调试平台,为学习好 C 语言提供了良好的基础;从第 2 章开始,每章内容组织形式全部采用“知识导例+相关知识+实践训练”的编写模式进行,读者学习时也按照该步骤,从“知识导例”中获取“疑问”,在“相关知识”中解决“疑问”、得到启发,在“实践训练”中强化知识并运用到实践中去,部分章节还增加了“知识拓展”,对不常用的语法规则或者较深的知识点在知识拓展中详细介绍,为读者较为全面地掌握 C 语言提供帮助。

本书共分 12 章,分别由河南工业职业技术学院的杜恒(第 1、8 章)、李森(第 2、10 章)、李金锁(第 4 章)、李垒(第 3、12 章)、王慧(第 6、11 章)、金梁(第 7 章)、方华丽(第 5、9 章)编写,参加编写的人员都有着较为丰富的编程经验,均在教学一线从事 C 语言教学多年。本书由杜恒任主编,李森任副主编,各章节标题的图标由河南工业职业技术学院于璐老师设计制作,全书由河南工业职业技术学院李怀刚副教授进行最后统稿与审定。

本书可以作为应用型本科、高职高专、中职等院校计算机及电子信息类专业学生学习 C 语言程序设计的教材,也可以作为全国计算机等级考试的重要参考书。同时,也是 C 语言学习爱好者的自学用书。本书内容翔实,知识点讲解全面,技能培养准确到位,在选用本书作为教材时,可根据各专业所制订的课程标准的要求或者根据学生的学习情况,对本书的内容进行取舍。

由于编者水平有限,对于编写 C 语言理论与实际一体化教程的经验还很欠缺,书中难免有错误、遗漏之处,恳请广大读者批评指正。

本书在编写过程中,得到了河南工业职业技术学院领导的大力支持,在此表示感谢。

编 者

目 录

前言

第1章 C语言概述.....1	
1.1 程序与算法.....1	
1.2 结构化程序设计方法.....5	
1.3 C语言程序初探.....7	
1.4 Visual C++ 6.0 开发平台简介.....11	
习题.....22	
第2章 数据类型、运算符及表达式.....24	
2.1 C语言中的标识符与数据类型.....24	
2.2 C语言中的基本数据类型、 常量与变量.....27	
2.3 算术运算符、自增自减运算符及其 相应表达式.....35	
2.4 关系、逻辑、条件运算符及其 相应表达式.....38	
2.5 赋值、逗号运算符及其相应表达式.....41	
2.6 混合运算及数据类型转换.....44	
2.7 综合实例.....46	
习题.....47	
第3章 顺序结构程序设计.....50	
3.1 赋值语句及数据的输出.....50	
3.2 数据的输入.....55	
3.3 复合语句与空语句.....58	
3.4 综合实训.....60	
习题.....62	
第4章 选择结构程序设计.....65	
4.1 if语句.....65	
4.2 if语句的嵌套.....68	
4.3 switch 多分支开关语句.....71	
4.4 goto 及语句标号.....75	
4.5 综合实训.....77	
习题.....78	

第5章 循环结构程序设计.....81	
5.1 while 语句.....81	
5.2 do-while 语句.....83	
5.3 for 语句.....86	
5.4 循环结构嵌套.....91	
5.5 break 语句与 continue 语句.....93	
5.6 综合实训.....96	
习题.....98	
第6章 数组.....104	
6.1 一维数组.....104	
6.2 二维数组.....108	
6.3 字符数组和字符串.....114	
6.4 综合实训.....121	
习题.....122	
第7章 函数.....128	
7.1 函数的定义及调用.....128	
7.2 函数调用时参数间的传递.....140	
7.3 函数的嵌套调用与递归调用.....147	
7.4 变量的作用域.....156	
7.5 变量的存储类别.....161	
7.6 内部函数与外部函数.....170	
7.7 综合实训.....173	
习题.....176	
第8章 指针.....182	
8.1 指针与指针变量.....182	
8.2 指针与一维数组.....189	
8.3 指针与二维数组.....195	
8.4 指针与字符串.....202	
8.5 指针数组与多级指针.....207	
8.6 函数指针与返回值为指针的函数.....212	
8.7 动态内存分配.....217	
8.8 命令行参数.....222	

8.9 综合实训.....	224	第 11 章 位运算.....	283
习题.....	228	11.1 常用位运算符及运算.....	283
第 9 章 编译预处理.....	232	11.2 综合实训.....	288
9.1 宏定义.....	232	习题.....	289
9.2 文件包含.....	235	第 12 章 文件操作.....	292
9.3 综合实训.....	238	12.1 文本文件操作.....	292
习题.....	239	12.2 二进制文件操作.....	299
第 10 章 结构体和共用体.....	242	12.3 文件的定位.....	303
10.1 结构体及结构体变量.....	242	12.4 综合实训.....	305
10.2 结构体数组.....	249	习题.....	308
10.3 结构体指针.....	252	附录.....	311
10.4 链表.....	259	附录 A 常用字符与 ASC II 代码对照表.....	311
10.5 共用体.....	269	附录 B 运算符优先级及结合性.....	313
10.6 综合实训.....	272	附录 C 常用库函数.....	314
习题.....	278	参考文献.....	322

第1章 C语言概述

计算机技术的发展日新月异, 计算机程序设计语言也层出不穷。语言是人与人、人与其他事物或者其他事物之间交流的工具, 计算机语言是人与计算机交流的工具, 本书介绍的 C 语言, 是一种结构化的、面向过程的高级语言, 它有着丰富的数据类型和运算符、灵活的语法和强大的功能。本章主要介绍程序、程序设计语言的相关概念、程序设计语言的发展历程、算法及算法的描述方法、结构化程序设计方法、C 语言程序的基本结构和执行流程, 还介绍了本书所使用的调试工具 Visual C++ 6.0 集成开发环境。

1.1 程序与算法

1. 程序与程序语言

(1) 计算机语言 什么是计算机语言? 一提到语言这个词, 人们自然想到的是汉语、英语等这样的自然语言, 因为它是人与人之间进行信息交流不可缺少的工具。同样在今天, 人和计算机进行信息交流时也需要使用语言, 这就是计算机语言。

(2) 程序与程序语言 计算机是一种以二进制数据形式在内部存储信息、以程序存储为基础、由程序自动控制的电子设备。人们需要计算机所做的任何工作, 都必须以计算机所能识别的指令形式送入计算机内存中。这种可以被连续执行的一条条有序指令的集合, 就称为程序。

更确切地说, 所谓程序, 是用计算机语言对所要解决的问题中的数据以及处理问题的方法和步骤所做的完整而准确的描述。对数据的描述就是指明数据结构; 对处理方法和步骤的描述就是算法。因而, 著名计算机科学家 Niklaus Wirth 教授提出一个著名的公式: 程序=数据结构+算法。

我们知道, 指令是二进制编码, 用它编制的程序很难掌握。所以, 计算机工作者就研制出了各种计算机能够执行、人们又方便使用的计算机语言, 程序就是用计算机语言编写的。因此, 计算机语言也被称为“程序语言”。

程序语言按语言级别可分为低级语言和高级语言。

计算机语言的发展也经历了多个发展阶段。

第一阶段, 是机器语言阶段, 也叫做二进制语言阶段。因为计算机作为一种电子产品, 是依靠电流工作的, 因此计算机也只能识别 0 (低电平) 和 1 (高电平) 组成的二进制指令。计算机发展初期, 科学家及计算机的使用者都是使用二进制语言进行编程, 当时编制程序的工具多数使用的是“纸带穿孔机”, 它的原理是在特制的黑色纸带上打上许多小孔, 然后用光电输入设备读取纸带, 同时用光线照射纸带, 有光线穿过的, 就有电信号产生, 读为 1; 没有光线穿过的, 没有电信号, 读为 0, 从而形成一连串的 0 和 1 构成的序列。

机器语言编程, 计算机不需要翻译就可以直接执行, 所以执行效率高, 但是编程效率极低, 人们往往很难记住那么多的二进制指令, 编程时需要不停地查阅工具书, 而且编制一个

较小的程序就可能需要很多的纸带。所以，编程效率极低，而且不适合大规模软件的开发。

第二阶段，是汇编语言。因为利用二进制语言进行编程，大量的二进制序列人们无法记忆，所以，科学家就将这些二进制序列用较容易记忆的一些单词、符号代替，使编程变得较为容易，这种语言叫做“助记符语言”。用这种语言编制出来的程序，必须经过一个称为“汇编”的编译工具，这种工具将助记符语言的一条语句或指令转换成机器能接受的二进制指令，转换的过程称为“汇编”。

例如下面一段代码片段，实现的是一个加法操作。

```
mov bl,ah
mov dl,al
add dl,30h
```

ah、al 表示 CPU 中的 ax 寄存器的高字节和低字节，add 表示执行加法指令。虽然助记符语言使用可以帮助记忆，但也只是稍微降低了编程的难度和门槛，使程序员在编制程序的时候，不至于总是一味地依靠工具资料。即便如此，汇编语言仍然难以普及，只在专业人员中使用，而且汇编语言可移植性较差，编译受机型的限制较大。汇编语言对于编程效率的提高没有质的变化，仍然难以编写大规模的软件，同时也不适合团队开发使用。

以上两种语言有一个共同的特点，都是非常依赖机器，在编程过程中，计算机的内存、寄存器等对于编程人员几乎“透明”，但是这些语言编制出来的程序计算机能够直接识别，或者经过简单编译即可翻译成计算机所需要的二进制代码，所以通常称这些贴近于计算机的语言为“低级语言”。

第三阶段，是面向过程的高级语言阶段。为了克服低级语言的这些弱点，在 20 世纪 50 年代，计算机科学家们开始研制适合于人类使用的，能用人类易懂的、贴近于人类自然语言的高级语言。Fortran 语言是第一个高级语言，它是由美国著名的计算机科学家约翰·巴克斯在 20 世纪 50 年代末设计出来的，约翰·巴克斯因此被誉为“Fortran 之父”。

在 Fortran 产生的几十年里，全世界涌现出 2 500 多种高级语言，其中广泛使用和流传下来的有 100 多种，很多已在计算机发展过程中被淘汰了。影响较大的语言如 Fortran、Basic、QBasic、Pascal、Cobol、PL/1、Lisp、C、C++、Java 等，多数都在目前广为流传使用。这些语言功能很强，具有很多优点，如不依赖于某个具体机器、使用贴近于人类的自然语言进行编程，因此称为高级语言。

高级语言从执行过程上分为两大类，一类是编译型的语言，另一类是解释型的语言。编译型的语言，是通过编译程序把用高级语言编制的源程序进行编译，生成机器能够识别的二进制目标文件，称为目标程序，然后再进行链接，将目标代码和所需要的函数等进行链接，生成可执行文件，再执行可执行文件得出结果，所以执行效率高。解释型的语言通过解释工具，边解释边执行，最后得到执行结果，所以解释型的程序执行效率较低。

在利用第三阶段的语言编制程序的时候，需要面向解决问题的每一个过程，考虑问题的每一个细节，代码重用性较小，所以也称这些语言为面向过程的高级语言。本书所介绍的 C 语言就是目前国内外广泛流行的面向过程的高级程序设计语言，它不仅可以用来编写系统软件，也可以用来编写应用软件。

第四阶段，是面向对象的高级语言。大规模软件的开发，使面向过程的高级语言逐渐暴露出不足之处，如开发效率较低，主要表现在代码的重用性不强；不太符合人类认识事物的规律，人类认识事物往往通过事物的个体入手，先认识整体，看到一个个对象，然后透过现象看本质，才认识事物的静态特征（属性）及动态行为。据此，科学家发明了面向

对象的高级语言。面向对象的语言将对象的属性和行为进行封装，封装成类，这符合事物的固有特征，而且大大提高了编程的效率，使规模化的开发成为可能，使软件工厂得以实现。面向对象程序设计语言的问世，在软件开发界产生了革命性的影响，使软件开发的规模和速度产生了质的飞跃。面向对象的程序设计产生于 20 世纪 70 年代，当时设计出的 Smalltalk 语言被公认为面向对象语言的鼻祖。现在常用的面向对象语言有 C++、Java、C# 等，其中 C++ 语言仍然支持面向过程的开发。

计算机的发展日新月异，同样程序设计语言的发展也是永不止步的，但无论怎样发展，目标只有一个，就是程序设计语言会越来越高级，开发软件会越来越方便，高度智能化的程序设计语言将是下一代计算机语言发展的目标。

2. 程序设计

程序设计就是使用某种程序语言编写程序的过程。

一般来说，程序设计应该包含以下 5 个步骤：

(1) 分析问题，建立模型 使用计算机解决具体问题时，首先要对问题进行充分的分析，确定解决问题的步骤。针对所要解决的问题，找出已知的数据和条件，确定所需的输入、处理及输出对象。对于多数数值类的科学计算问题，可以将解题过程归纳为一系列的数学表达式，建立各种量之间的关系，即建立起解决问题的数学模型；对于有些数据，不一定能得出其数学公式，但是可以通过分析建立非数学模型。

(2) 确定数据结构和算法 根据建立的模型，规划输入的数据和预期输出的数据，确定存放数据的数据结构，并选择合适的算法加以实现。算法是指为解决某一特定问题而采取的确定的有限的步骤。对同一问题，每一个人确定的算法都可能有所不同。

(3) 编写程序 根据确定的数据结构和算法，使用某种程序设计语言把这个解决问题的方案按照严格的语法规则描述出来，就是编写程序。

(4) 调试程序 程序开发人员编写的程序称为源程序或源代码，源代码不能被计算机执行。源代码要经过编译程序编译，生成目标程序，然后链接其他相应的代码，最后生成可被计算机执行的可执行文件。在程序调试过程中，要不断分析所得到的运行结果，进行程序的测试和调整，直至获得预期的结果。

(5) 运行测试 程序编写、调试完毕，得出运行结果，必须要进行测试。测试是发现问题的过程，发现问题越多，说明测试的效果越好。一个程序测试没有发现问题，并不等于该程序没有任何问题，可能是没有发现问题，或者是由于设计的测试用例不科学等原因造成。所以一般在程序开发完毕进行测试时，都由第三方或者软件企业中独立的测试部门进行测试，也可以邀请用户进行测试，以求达到最佳的测试效果。

3. 算法

(1) 算法的基本概念 在日常生活中解决任何问题的过程都是由一定的方法和步骤组成的，这些确定的方法和有限的步骤就是算法。

例如，计算圆的面积。算法可以书写如下。

第一步：给出圆的半径 r 。

第二步：计算出圆的面积 s 。

第三步：输出圆的面积 s 。

求圆的面积问题，由于步骤非常简单，而且存在着一个已经发现的数学模型，即数学公式 $s = \pi r^2$ ，所以问题很好求解。

再如，求 $1+2+3+\cdots+99+100$ 的值。

算法 1:

$$1+2=3, 3+3=6, 6+4=10, \cdots, 4\ 851+99=4\ 950, 4\ 950+100=5\ 050$$

算法 2:

$$100+(1+99)+(2+98)+\cdots+(48+52)+(49+51)+50=100+49\times 100+50=5\ 050$$

当然还可以有其他的算法。很明显，算法有优劣之分，一般希望采用简单的和运算步骤少的方法。由此可以看到，对同一个问题，也可以有不同的解题方法和步骤。为了有效地解决问题，不仅需要保证算法正确，还要考虑算法的质量，选择尽可能合适的算法。

以上介绍的两个例子均是可以建立数学模型来解决的问题，对于非数学模型，在此不做探讨。

(2) 算法的基本特征 一个算法应该具有以下基本特征：

- 1) 有穷性。一个算法必须在执行有限个操作步骤后终止。
- 2) 确定性。算法中的每一个步骤都应该是唯一的和确切的，不可出现二义性。
- 3) 有零个或多个输入。在执行算法时，需要从外界得到的必要信息就是输入。有些特殊算法也可以没有输入。
- 4) 有一个或多个输出。算法的目的是为了求解，算法得到的结果就是该算法的输出，没有输出的算法是没有意义的。一个算法可以有一个或多个与输入相关的输出。
- 5) 有效性。算法的每一步操作都应该能有效执行，并得到确定的结果，一个不可执行的操作是无效的。例如， $b=0$ 时， a/b 是不能有效执行的，应当避免这种操作。

4. 算法的表示

为了表示一个算法，可以使用不同的方法。通常可使用的方法有自然语言、传统流程图、N-S 结构化流程图、PAD 图、伪代码等。

(1) 自然语言 自然语言就是人们日常使用的语言，可以是汉语、英语或其他语言。用自然语言表示通俗易懂，但文字冗长，可能还会出现语义表达上的问题等。

(2) 传统流程图 所谓流程图，就是用一些图框表示各种操作。用图形表示算法，其优点是形象直观、易于理解、便于修改和交流。美国标准化协会（ANSI）规定了一些常用的流程图符号（见表 1-1），已被世界各国的广大程序设计工作者普遍接受和采用。采用这些流程图符号所表示的流程图称为传统流程图，又称为一般流程图。

表 1-1 一般流程图所使用的标准符号

符号名称	符 号	功 能
起止框		表示算法的开始和结束
输入输出框		表示算法的输入/输出操作，框内填写需输入/输出的各项
处理框		表示算法中的各种处理操作，框内填写处理说明或算式
判断框		表示算法中的条件判断操作，框内填写判断条件
注释框		表示算法中某操作的说明信息，框内填写文字说明
流程线		表示算法的执行方向
连接点		表示流程图的延续

例如计算 $s=1+2+3+\cdots+99+100$ 的值,用传统流程图表示,算法如下:

首先,确定变量 s 的初始值为 0,变量 i 的初始值为 1。

其次,确定公式 $s=s+i$ (这里的“=”不同于数学里的等号,它表示赋值)。

当 i 分别取值 1, 2, 3, 4, $\cdots$, 99, 100 时,重复计算公式 $s=s+i$, 计算 100 次后,即可求出 s 的最终结果。

用传统流程图表示,如图 1-1 所示。

(3) N-S 结构化流程图 1973 年,两位美国学者 Nassi 和 Shneiderman 又提出了一种新的流程图形式,这就是 N-S 结构化流程图 (简称 N-S 流程图)。N-S 流程图是将算法的每一个步骤,按序连接成一个大的矩形框来表示,矩形框由三部分构成,最上面是输入,最下面是输出,中间是流程的主要部分,可以任意扩展,从而完整地描述一个算法。N-S 图由于从整体上看是一个大的矩形,像一个盒子,所以又称为盒图。用 N-S 流程图表示 $1+2+3+\cdots+99+100$ 的问题求解算法如图 1-2 所示。

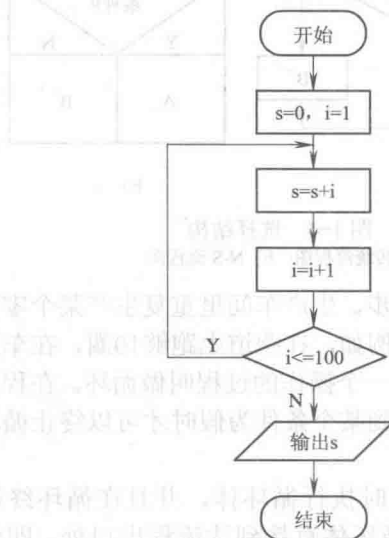


图 1-1 传统流程图

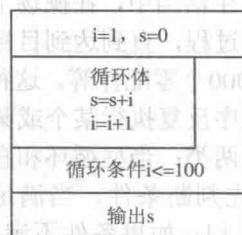


图 1-2 N-S 流程图

1.2 结构化程序设计方法

1. 程序的三种基本结构

结构化程序的概念首先是从以往编程过程中无限制地使用无条件转移语句 (goto 语句) 而提出的。如果一个程序中多次出现转移情况,将会导致程序流程无序可寻,程序结构杂乱无章,阅读者要花很大精力去理解流程,分析算法的逻辑,算法的可靠性和可维护性难以保证。结构化语言比非结构化语言易于程序设计,使用结构化语言编写的程序清晰易懂,便于维护。

1966 年,计算机科学家 Bohm 和 Jacopini 提出了程序设计的三种基本结构,也是结构化程序设计必须采用的结构。

(1) 顺序结构 顺序结构表示程序中的各操作是按照它们出现的先后顺序执行的,其流程如图 1-3 所示。图中的 A 和 B 表示两个处理步骤,整个顺序结构只有一个入口点和一个出口点。这种结构的特点是:程序从入口点开始,按顺序执行所有操作,直到出口点处,所以称

为顺序结构。顺序结构是最简单的一种基本结构。

(2) 选择结构 生活当中充满了选择，人的一生每一步都在进行选择。例如，走到岔路口要选择应该走哪一条路，去餐厅吃饭要选择吃什么饭等。在自然界和社会生活中，选择结构是非常常见的。选择结构有一个特点，就是在做某件事情之前就要进行判断，放在程序中就是要书写判定条件。选择结构表示程序的处理步骤中出现了分支，需要根据某一特定的条件选择其中的一个分支执行，其流程如图1-4所示。在图1-4中，用P表示需要判断的条件，用Y表示条件为真，用N表示条件为假，用A和B表示流程的两个分支，分支由语句或语句块组成的。由图1-4可见，在结构的入口点处是一个判断框，表示程序流程出现了两个可供选择的分支，如果条件满足执行A处理，否则执行B处理。在这两个分支中只能选择一个且必须选择一个执行，但不论选择了哪一个分支执行，最后流程都一定到达结构的出口点处。

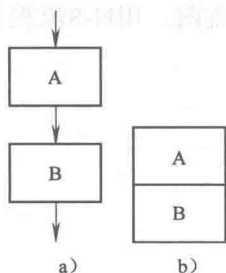


图 1-3 顺序结构
a) 传统流程图 b) N-S 流程图

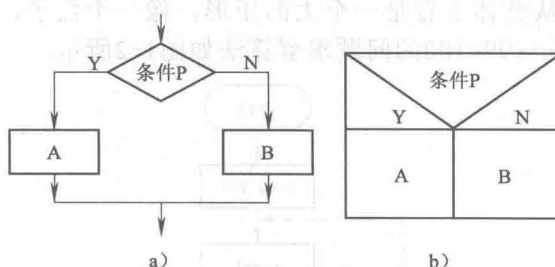


图 1-4 选择结构
a) 传统流程图 b) N-S 流程图

(3) 循环结构 生活当中，在操场上一圈一圈地跑步，生产车间里重复生产某个零配件等都是重复某一个既定过程，直到达到目标要求为止。例如，在跑道上跑够10圈，在车间里生产完成客户需要的1000个零配件等。这种重复执行某一个操作的过程叫做循环。在程序设计中，循环结构表示程序反复执行某个或某些操作，直到某个条件为假时才可以终止循环。

循环结构又分为两类：当型循环和直到型循环。

1) 当型循环。先判断条件，当满足给定的条件时执行循环体，并且在循环终端处流程自动返回到循环入口；如果条件不满足，则退出循环体直接到达流程出口处，即先判断后执行。其流程如图 1-5 所示。

2) 直到型循环。从结构入口处直接执行循环体，在循环终端处判断条件，如果条件满足，返回入口处继续执行循环体，直到条件为假时再退出循环到达流程出口处，是先执行后判断。其流程如图 1-6 所示。

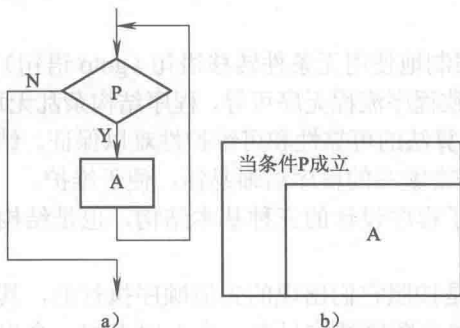


图 1-5 当型循环
a) 传统流程图 b) N-S 流程图

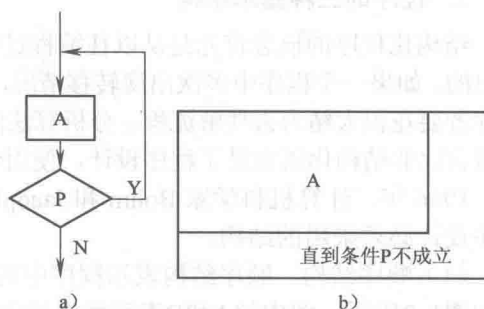


图 1-6 直到型循环
a) 传统流程图 b) N-S 流程图

2. 结构化程序设计方法

(1) 结构化程序设计特征 结构化程序设计主要有以下几个特征。

- 1) 采用自顶向下, 逐步求精的编程思想来进行整体设计。
- 2) 以三种基本结构的组合来描述程序。
- 3) 整个程序的编写采用模块化的组织方法。
- 4) 采用结构化程序设计语言书写程序, 并使程序清晰、易读。

(2) 结构化程序设计方法 结构化程序设计及方法有一整套不断发展和完善的理论和技巧, 对于初学者来说, 完全掌握是比较困难的。但在学习的起步阶段就了解结构化程序设计的方法, 学习好的程序设计思路, 对今后的实际编程是很有帮助的。

一个结构化程序就是用高级语言表示的结构化算法。结构化算法设计的总体思路是采用模块化结构, 自顶向下, 逐步求精。即首先把一个复杂的大问题的求解过程分解为若干个相对独立的小问题, 如果小问题仍然比较复杂, 则可以把这些小问题进一步分解成若干个子问题, 这样不断地分解, 使得小问题或子问题简单到能够直接用程序的三种基本结构表达为止。然后, 对应每一个小问题或子问题编写出一个功能上相对独立的程序块, 这些程序块称为模块。每个模块完成之后, 最后再统一组装, 这样, 对一个复杂问题的解决就变成了对若干个简单问题的求解。这就是自顶向下、逐步求精的程序设计方法。

由此可见, 模块是程序对象的集合, 模块化就是把一个程序划分成若干个模块, 每个模块完成一个确定的功能, 把这些模块集中起来组成一个整体, 就可以完成对问题的求解。这种用模块组装起来的程序称为模块化结构程序。在模块化结构程序设计中, 采用自顶向下、逐步求精的设计方法便于对问题的分解和模块的划分。所以, 它是结构化程序设计的基本原则。

基于结构化编程思想、采用三种固定的程序结构和模块化的程序设计思想这三条原则设计的程序设计语言, 称为结构化程序设计语言。C语言就是一种结构化程序设计语言。结构化程序设计语言非常多, 如 PASCAL、QBASIC、FORTRAN77 等。

1.3 C语言程序初探

1. C语言的发展

C语言是国际上广泛流行的、很有发展前途的计算机高级程序设计语言。它既适合于作为系统描述语言, 又可用于编写应用软件。

对C语言的研究起源于系统程序设计的深入研究和发 展。1967年, 英国剑桥大学的 M.Richards 在 CPL 语言的基础上实现并推出了 BCPL (Basic Combined Programming Language) 语言。1970年, 美国贝尔实验室的 Ken Thompson 以 BCPL 语言为基础, 又进一步简化设计出了 B 语言, 并用 B 语言写了第一个 UNIX 操作系统。由于 B 语言过于简单, 功能有限, 1972年, 贝尔实验室的 Dennis M.Ritchie 在 B 语言的基础上设计出了 C 语言。1973年, 贝尔实验室的 K.Thompson 和 Dennis M.Ritchie 合作, 用 C 语言重新改写了 UNIX 操作系统。此后, 伴随着 UNIX 操作系统的发展, C 语言越来越广泛地被人们接受和应用。

至此, C 语言不断得到改进, 但主要还是作为实验室产品在使用, 因为它仍然依赖于具体的机器。直到 1977 年才出现了独立于具体机器的 C 语言编译版本。1978 年, Brian W.Kernighan 和 Dennis M.Ritchie 正式出版了影响深远的《The C Programming Language》一

书,此书中介绍的C语言成为后来广泛使用的C语言版本基础,被称为标准C语言。

C语言的标准化工作是从20世纪80年代初期开始的。1983年,美国国家标准化协会(ANSI)颁布了C语言的新标准ANSI C。由于C语言的不断发展,1987年,美国国家标准化协会又颁布了新标准87 ANSI C。1990年,87 ANSI C成为ISO C的标准,目前流行的C编译系统都是以它为基础的,如Borland公司的Turbo C、Microsoft公司的Microsoft C和Visual C++等,这些编译系统在我国使用也十分广泛。

1999年,C语言颁布了最新的标准C99。

2. C语言的特点

C语言是目前世界上计算机广泛应用的一种高级程序设计语言,具有较强的生命力。C语言之所以能够存在且持续发展,正是由于它自身具备的突出特点。

C语言的主要特点如下:

1) 语言简洁、使用方便。C语言一共只有32个关键字,程序书写自由。与其他语言相比,C语言的书写形式更为直观、精炼。

2) 适应性强、应用范围广。C语言能够适应从8位微型机到巨型机的所有机种,并可应用于系统软件以及各个领域的应用软件。

3) 运算符丰富,语言的表达能力强。C语言共有34种运算符,把括号、赋值号、强制类型转换等作为运算符处理,使其运算类型丰富、灵活、多样,功能强大,可直接处理字符,访问内存地址,进行位操作等。

4) 语法限制不严格,程序设计灵活自由。一个比较明显的语法规则就是数组的越界问题,在C语言中,数组越界,编译系统并不给出错误提示,而是由设计者自己去处理。这些问题给初学者带来不便,但是对于掌握了C语言的编程人员则有好处,因为设计程序灵活,自由度高,而且能实现较强的功能。

5) 数据结构系统化。C语言具有现代化语言的各种数据结构,且具有数据类型的构造能力,因此,便于实现各种复杂的数据结构的运算。

6) 具有结构化的控制语句。C语言是结构化的程序设计语言,提供了各种控制语句(如if、while、for、switch等),对程序的逻辑结构提供了很好的基础。其程序结构清晰,层次分明,有利于采用自顶向下、逐步求精的程序设计方法。

7) 具有低级语言的某些特征。C语言通过指针或位运算,能使程序直接访问到内存的物理地址,甚至能对某个存储单元的二进制位进行操作,而这些正是低级语言所具备的功能。也正是C语言的这种双重特征,使C语言功能更加强大,不仅可以开发系统软件,也可以开发应用软件。

8) 运行程序质量高,程序运行效率高。试验表明,C语言源程序生成的运行程序的效率仅比汇编程序的效率低10%~20%。

9) 可移植性好。C语言编译程序基本上不做修改就能用到各种型号的计算机和各种操作系统上。

3. C语言程序的基本结构

用C语言编写的程序称为C语言程序或C语言源程序。下面,通过一个简单的C语言程序实例来认识一个完整的C语言程序的基本结构。

该程序的功能是求两个数的最大值,其中求两个数的最大值定义成一个函数,在主函数

中进行调用。

```
#include "stdio.h"          /*包含标准的输入输出头文件*/
int max(int x,int y);        /*函数声明*/
main ()                      /*主函数*/
{
    int a,b,c;               /*声明部分,定义a,b,c为整型变量*/
    scanf("%d,%d",&a,&b);    /*由键盘输入a和b的值*/
    c=max(a,b);              /*调用max函数,将得到的值赋给c*/
    printf("max=%d",c);      /*输出c的值*/
}
int max(int x,int y)         /*定义max函数*/
{
    int z;                   /*在max函数中定义变量z为整型*/
    if(x>y) z=x;             /*如果x>y,将x赋值给z,否则将y赋值给z*/
    else z=y;                /*将z的值返回到函数的调用处*/
    return(z);
}
```

这个程序在执行时,如在屏幕上输入3,5,则在屏幕上输出: max=5。

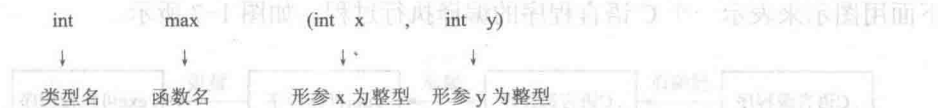
说明:

- 1) 本程序包含两个函数: main 函数和 max 函数。
 - 2) 由一对花括号“{}”括起来的是函数体。
 - 3) 在 main 函数中通过赋值语句“c=max(a,b);”调用 max 函数。
 - 4) max 函数的功能是将 x 和 y 中较大者的值赋给变量 z, return 语句将 z 的值返回给主调函数 main。
 - 5) 程序开头使用“#include "stdio.h"”宏定义命令包含了标准输入输出头文件,因为在程序中,用到了 stdio.h 头文件中的输入函数 scanf 和输出函数 printf。
 - 6) max 定义在调用它的函数后面,所以在调用之前必须进行声明。本程序把声明语句 int max(int x,int y); 放在了主调函数的前面,也可以放在调用的语句前面。
- 一般来说,一个完整的 C 语言程序结构有以下特点:

1) C 语言是由函数构成的。函数是 C 语言程序的基本单位。其中 main 函数是一个特殊的函数,一个完整的 C 语言程序必须有且仅有一个 main 函数,它是程序启动时的唯一入口(不论 main 函数的位置如何)。除 main 函数之外, C 语言程序也可以包含若干个其他 C 语言标准函数和用户自定义的函数,它们可以相互调用,并最终返回给主函数。

2) 一个函数是由函数头和函数体两个部分组成。

函数头主要包括函数的类型、函数名、形式参数的类型、个数及形式参数名称。如上例中的 max 函数:



函数体是由函数头下面的一对花括号“{}”内的一系列语句和注释构成。函数体包括变量的定义和可执行语句两部分。

变量定义: 如 main 函数中的“int a,b,c;”语句, max 函数中的“int z;”语句。

可执行语句：完成当前函数功能的语句。

3) 程序书写格式较自由，一行可写几条语句，一条语句也可以分行书写。

4) 语句以分号结束，如“z=y;”。

5) 在每条语句后，可用/* … */对该语句进行注释，中间的省略号表示注释的内容。该注释方式可以注释多行，以增加程序的可读性。注释不影响语句的功能。本书所使用的 VC++ 6.0 平台中，注释还有另一种写法，即使用“//”符号。使用“//”注释方式只能注释单行。

4. C 语言程序的执行流程

C 系列的语言是编译型的语言，C 语言是其中的一种。用 C 语言编写的源程序必须要经过编译和链接这两个最主要的阶段，才能最终生成可以执行的文件，执行效率非常高。编译型的语言有别于解释型的语言，解释型的语言对于写好的源程序，边解释边执行，所以执行效率较低。

一般来讲，一个完整的 C 语言程序从开始编辑到最后生成可执行文件要经过这样几个步骤。

(1) C 语言程序编辑 通过编辑器编辑好的程序称为源程序。C 语言源程序可以通过文本编辑器或者通过 TurboC 2.0、Visual C++ 6.0 等编译系统的编辑器来进行编写。编写完的源程序是文本文件，但是其扩展名必须定义为“.C”，这样才是 C 语言的源程序文件。

(2) C 语言程序源文件的编译 编辑完的 C 语言源程序可以通过编译系统进行编译，编译的主要功能对源程序进行词法分析、语法检查等，如检查标识符的定义是否合法；检查程序的三种基本结构是否正确；语句是否正确；检查函数定义与调用是否正确等。通过编译的 C 语言源程序生成一个后缀名是“.obj”的目标文件，该文件是一个二进制文件。

在编译阶段需要注意，如果 C 语言的源程序里面用到了 include、define 等预编译命令时，那么在 C 语言源程序编译之前，又会多出一个阶段，叫做预编译。预编译的主要功能是进行内容的替换，该阶段并不进行语法检查。该阶段因为它位于编译之前，所以称为预编译。在多数编译系统中一般都没有单独的预编译命令菜单，此命令往往是和编译合在一起的，但是执行过程仍然是按照先预编译再编译这个顺序执行的。

(3) C 语言程序目标文件的链接 对于编译通过的 C 语言目标文件，通过编译系统的链接命令，可以进行目标文件和 C 语言的标准库函数以及用户自定义函数等进行链接，链接成功后，将生成一个扩展名是“.exe”的可执行文件。如果在链接过程中，没有相应的函数，则链接不能进行，也无法生成可执行文件。

(4) C 语言程序可执行文件的运行 通过链接生成的扩展名是“.exe”的文件，可以在 DOS 控制台中通过输入可执行文件的文件名执行，也可以在 Windows 可视化窗口中通过鼠标点击执行。可执行文件执行后，会按照程序的要求输出相应结果。

下面用图示来表示一个 C 语言程序的编译执行过程，如图 1-7 所示。



图 1-7 C 语言程序的编译执行过程

1.4 Visual C++ 6.0 开发平台简介

C 语言的调试工具非常多,常用的有 Borland 公司的 Turbo C 系列和 MicroSoft 公司的 Visual C++ 系列。本书使用 Visual C++ 6.0 集成开发环境。

Visual C++ 6.0 是一个可视化的编程工具,功能非常强大,尤其是调试功能更是其他语言调试工具无法相比的。

1. Visual C++ 6.0 集成开发环境的启动

如果计算机中已经安装了 Visual Studio 6.0,可以通过在开始菜单中“开始→所有程序→Microsoft Visual Studio 6.0→Visual C++ 6.0”步骤打开 Visual C++ 6.0 集成开发环境,如图 1-8 所示。启动后的 Visual C++ 6.0 集成开发环境如图 1-9 所示。

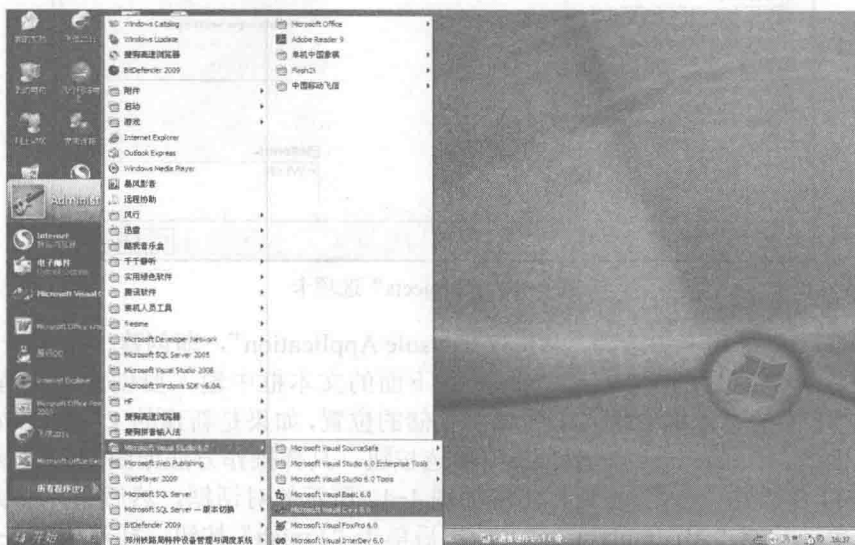


图 1-8 准备启动 C 语言的编译环境 Visual C++ 6.0

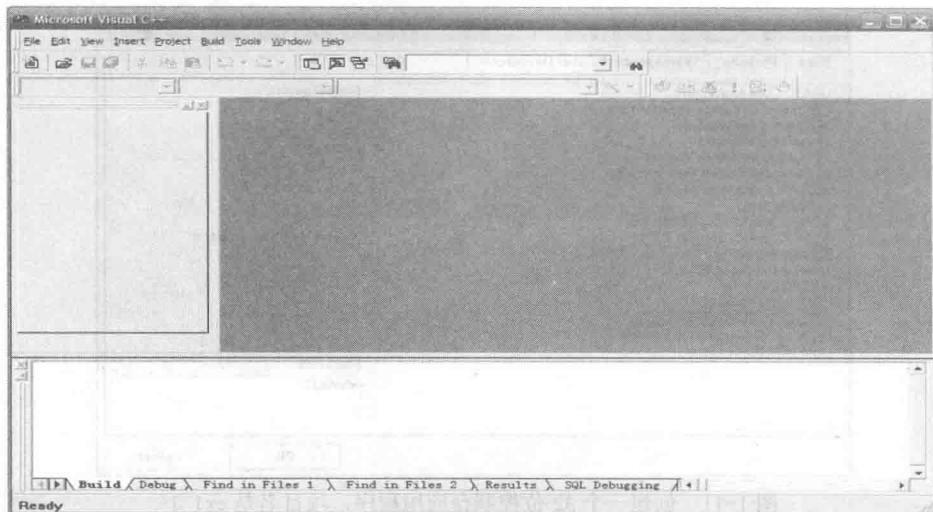


图 1-9 Visual C++ 6.0 集成开发环境