

高等院校计算机应用系列教材

C 语言程序设计

(第三版)(微课版)

		梁海英	董延华	主 编
姚建盛	李淑梅	罗 琳	千 文	副主编

清华大学出版社
北 京

内 容 简 介

本书按照程序设计的体系结构,系统介绍了C语言程序设计的基本思想及基本方法。全书内容分为12章。第1~7章介绍了C语言程序设计的基本应用,包括:C语言程序的结构,数据类型及基本运算量,结构化程序设计的顺序结构、选择结构、循环结构,数组及函数的开发方法与应用实现;第8~12章详细介绍了C程序设计的高级应用,包括:编译预处理、结构体、共用体、指针、文件操作及位运算等应用。本书最后的附录提供了全国计算机等级考试二级C语言程序设计考试大纲及模拟题。

本书可作为高等院校计算机类相关专业的程序设计入门教材或高等院校非计算机专业本科生的计算机通识课教材,也可作为全国计算机等级考试的参考用书,还可作为ACM和“蓝桥杯”等IT类学科竞赛的参考用书。

本书配套的电子课件、习题答案和实例源文件可以到<http://www.tupwk.com.cn/downpage> 网站下载,也可以扫描前言中的二维码获取。扫描前言中的“看视频”二维码可以直接观看教学视频。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。举报:010-62782989, beiqinquan@tup.tsinghua.edu.cn

图书在版编目(CIP)数据

C语言程序设计:微课版/梁海英,董延华主编. —3版. —北京:清华大学出版社,2023.3
高等院校计算机应用系列教材
ISBN 978-7-302-63084-5

I. ①C… II. ①梁… ②董… III. ①C语言—程序设计—高等学校—教材 IV. ①TP312.8

中国国家版本馆 CIP 数据核字(2023)第 043429 号

责任编辑:胡辰浩

封面设计:高娟妮

版式设计:妙思品位

责任校对:成凤进

责任印制:刘海龙

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦A座 邮 编:100084

社总机:010-83470000 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

印装者:北京国马印刷厂

经 销:全国新华书店

开 本:185mm×260mm 印 张:19.25 字 数:493千字

版 次:2013年1月第1版 2023年5月第3版 印 次:2023年5月第1次印刷

定 价:79.00元

产品编号:098656-01

前 言

我们基于多年的丰富教学经验及素材积累，精心编写了此书，目的是让初学者能够循序渐进地掌握程序设计的思想，系统地掌握 C 语言程序设计的方法。本书从实用的角度出发，选取适当的相关案例，配备简洁的讲解文字，辅助直观的算法流程图，编写缩进格式的实现程序，插入真实有效的运行结果。本书针对初学者的特点和认知规律，精选内容，分散难点，降低台阶，丰富例题，深入浅出。

全书内容共分 12 章：第 1 章介绍 C 语言程序结构及其特点；第 2 章介绍数据类型、常量、变量、库函数和表达式；第 3 章介绍用传统流程图及 N-S 结构化流程图实现结构化程序设计的三种基本结构、赋值语句、数据输入/输出函数调用语句及顺序结构程序设计的方法；第 4 章介绍关系运算符和关系表达式、逻辑运算符和逻辑表达式、用 if 语句和 switch 语句实现选择结构程序设计的方法；第 5 章介绍用 while 语句、do-while 语句和 for 语句实现循环结构程序设计，以及用 break 和 continue 语句提前结束循环的方法；第 6 章介绍数组的定义和初始化、数组元素的使用、数值数组元素的常见操作、字符数组的使用方法；第 7 章介绍函数的定义、被调函数的声明、函数的调用、数组作为函数参数、变量的作用域和存储类别；第 8 章介绍宏定义、文件包含；第 9 章介绍结构体类型、共用体类型、枚举类型；第 10 章依次介绍指向变量的、指向数组的、指向函数的、指向指针的和指向结构体的指针变量及动态存储分配；第 11 章介绍文件的打开与关闭、文件的顺序读写和随机读写及文件检测函数；第 12 章介绍位运算符和位域。另外，本书最后的附录提供了全国计算机等级考试二级 C 语言程序设计考试大纲及模拟题。

本书为广西教育厅高等学校科研项目(ZD2014129)研究成果，由贺州学院梁海英和吉林师范大学董延华任主编，桂林理工大学姚建盛、吉林师范大学李淑梅、吉林师范大学罗琳和贺州学院千文任副主编，全书由梁海英统稿。在本书的编写过程中，得到了作者所在学院同事们热心帮助和支持，参加本书编写工作的老师还有陈冠萍、袁淑丹、庄兴义、马文成、樊艳英、王雪红、肖鸿、罗志林、刘艳玲等。在此，向他们表示衷心的感谢！

在本书的编写过程中参考了相关文献，在此向这些文献的作者深表感谢。由于作者水平有限，书中难免有不足之处，恳请专家和广大读者批评指正。我们的电话是 010-62796045，信箱是 992116@qq.com。

本书配套的电子课件、习题答案和实例源文件可以到 <http://www.tupwk.com.cn/downpage> 网站下载，也可以扫描下方二维码获取。扫描下方二维码“看视频”二维码可以直接观看教学视频。

配套资源



扫描下载

扫一扫



看视频

编 者
2022 年 10 月

目 录

第 1 章 引言	1
1.1 程序设计语言	1
1.1.1 低级语言	1
1.1.2 高级语言	2
1.2 程序结构及其特点	2
1.2.1 程序结构	3
1.2.2 程序结构的特点	3
1.2.3 程序书写规则	4
1.2.4 程序保留字	5
1.3 Visual C++ 6.0 的安装及使用	6
1.3.1 Visual C++ 6.0 的安装	6
1.3.2 Visual C++ 6.0 的启动	6
1.3.3 Visual C++ 6.0 上机过程	7
1.3.4 Visual C++ 6.0 的退出	9
1.4 习题	9
第 2 章 数据类型及基本运算量	11
2.1 数据类型	11
2.1.1 基本数据类型	11
2.1.2 构造数据类型	16
2.1.3 指针类型	16
2.1.4 空类型(void)	16
2.2 常量	16
2.2.1 直接常量	17
2.2.2 符号常量	18
2.3 变量	20
2.3.1 变量的定义	20

2.3.2 变量的使用	21
2.4 库函数	22
2.4.1 数学函数	22
2.4.2 输入/输出函数	23
2.5 运算符及表达式	23
2.5.1 运算符及表达式简介	24
2.5.2 算术运算符和算术表达式	27
2.5.3 赋值运算符和赋值表达式	31
2.5.4 逗号运算符和逗号表达式	33
2.6 习题	34
第 3 章 顺序结构程序设计	37
3.1 结构化程序设计概述	37
3.1.1 结构化程序的特点	37
3.1.2 结构化程序的设计方法	37
3.2 用流程图表示 3 种基本结构	38
3.3 C 语句概述	41
3.4 赋值语句	42
3.5 数据输入/输出函数调用语句	43
3.5.1 数据输入/输出的方法	43
3.5.2 字符输入/输出	43
3.5.3 格式输入/输出	45
3.6 顺序结构程序设计举例	51
3.7 习题	52
第 4 章 选择结构程序设计	55
4.1 选择结构程序设计概述	55
4.2 关系运算符和表达式	56

4.2.1 关系运算符.....	56	6.5.1 一维数组元素的常见操作.....	109
4.2.2 关系表达式.....	56	6.5.2 二维数组元素的常见操作.....	122
4.3 逻辑运算符和表达式.....	57	6.6 数值数组的应用举例.....	128
4.3.1 逻辑运算符.....	57	6.6.1 一维数组程序举例.....	128
4.3.2 逻辑表达式.....	58	6.6.2 二维数组程序举例.....	132
4.4 用if语句实现选择结构程序设计.....	59	6.7 字符数组的使用.....	133
4.4.1 if语句的3种形式.....	60	6.7.1 字符串和字符串结束标志.....	134
4.4.2 if语句的嵌套.....	64	6.7.2 字符数组的输入/输出.....	134
4.4.3 条件运算符和条件表达式.....	67	6.7.3 字符串处理函数.....	135
4.5 用if语句实现选择结构程序设计.....	68	6.8 字符数组应用程序举例.....	140
4.6 选择结构程序设计举例.....	71	6.9 习题.....	141
4.7 习题.....	73		
第5章 循环结构程序设计.....	77	第7章 用户自定义函数.....	147
5.1 循环结构程序设计概述.....	77	7.1 用户自定义函数的种类.....	147
5.2 用于实现循环结构程序设计的 语句.....	77	7.2 函数的定义.....	148
5.2.1 用while语句实现循环结构 程序设计.....	77	7.3 被调函数的声明.....	150
5.2.2 用do-while语句实现循环结构 程序设计.....	82	7.4 函数的调用.....	151
5.2.3 用for语句实现循环结构程序设计.....	85	7.4.1 函数调用的一般形式.....	151
5.2.4 循环的嵌套.....	88	7.4.2 函数调用的方式.....	151
5.2.5 几种循环语句的比较.....	90	7.4.3 函数调用的参数传递.....	152
5.3 用break和continue语句 提前结束循环.....	90	7.5 函数的嵌套调用.....	154
5.3.1 break语句.....	91	7.6 函数的递归调用.....	155
5.3.2 continue语句.....	92	7.7 数组作为函数参数.....	157
5.4 循环结构程序设计举例.....	93	7.7.1 数组元素作为函数实参.....	157
5.5 习题.....	96	7.7.2 数组名作为函数参数.....	158
		7.8 变量的作用域.....	160
第6章 数组.....	101	7.8.1 局部变量.....	160
6.1 数组的概念.....	101	7.8.2 全局变量.....	161
6.2 数组的定义.....	102	7.9 变量的存储类别.....	163
6.3 数组的初始化.....	104	7.9.1 静态存储方式的变量与 动态存储方式的变量.....	163
6.4 数组元素的使用.....	106	7.9.2 用auto声明动态局部变量.....	163
6.5 数值数组元素的常见操作.....	109	7.9.3 用static声明静态局部变量.....	164
		7.9.4 用register声明寄存器变量.....	165
		7.9.5 用extern声明外部变量.....	166
		7.10 习题.....	167

第 8 章 编译预处理	173		
8.1 宏定义	173		
8.1.1 无参宏定义	173		
8.1.2 带参宏定义	175		
8.2 文件包含	180		
8.3 习题	181		
第 9 章 用户自定义数据类型	183		
9.1 结构体类型	183		
9.1.1 结构体类型的定义	183		
9.1.2 结构体类型变量的定义	184		
9.1.3 结构体类型变量的成员 变量的表示方法	185		
9.1.4 结构体类型变量的 成员变量的用法	186		
9.1.5 结构体类型变量的初始化及 整体赋值	187		
9.1.6 结构体类型数组的定义和使用	187		
9.2 共用体类型	189		
9.2.1 共用体类型的定义	189		
9.2.2 共用体类型变量的使用	189		
9.3 枚举类型	191		
9.3.1 枚举类型的定义	191		
9.3.2 枚举类型变量的定义	191		
9.3.3 枚举类型变量的使用	192		
9.4 类型声明符 typedef	194		
9.5 习题	195		
第 10 章 指针	199		
10.1 指针的基本概念	199		
10.2 指向变量的指针变量	200		
10.2.1 指针变量的定义	200		
10.2.2 指针运算符	200		
10.2.3 指针变量作为函数参数	203		
10.3 指向数组的指针变量	206		
10.3.1 指向数组的指针变量的 定义与赋值	206		
10.3.2 通过指针变量引用数组元素	207		
10.3.3 指向数组的指针变量作为 函数参数	211		
10.3.4 多维数组的地址和指向 多维数组的指针变量	214		
10.3.5 字符串的地址和指向字符串的 指针变量	217		
10.4 指向函数的指针变量和 指针型函数	220		
10.4.1 指向函数的指针变量	220		
10.4.2 指针型函数	222		
10.5 指针型数组和指向指针的 指针变量	223		
10.5.1 指针型数组的定义及使用	223		
10.5.2 指向指针的指针变量	227		
10.6 指向结构体的指针变量	228		
10.6.1 指向结构体变量的指针变量	229		
10.6.2 指向结构体数组的指针变量	230		
10.6.3 结构体指针变量作为函数参数	231		
10.7 动态存储分配	233		
10.8 习题	235		
第 11 章 文件	241		
11.1 文件的种类	241		
11.2 文件指针和文件内部的 位置指针	242		
11.3 文件的操作	242		
11.4 文件的打开与关闭	244		
11.4.1 文件打开函数 fopen	244		
11.4.2 文件关闭函数 fclose	246		
11.5 文件的顺序读写	246		
11.5.1 字符读写函数 fgetc 和 fputc	246		
11.5.2 字符串读写函数 fgets 和 fputs	248		
11.5.3 数据块读写函数 fread 和 fwrite	250		
11.5.4 格式化读写函数 fscanf 和 printf	252		
11.6 文件的定位和随机读写	254		
11.6.1 文件的定位	254		

11.6.2 文件的随机读写	255	12.1.5 左移运算	261
11.7 文件检测函数	256	12.1.6 右移运算	262
11.7.1 文件结束检测函数feof	256	12.2 位域(位段)	262
11.7.2 读写文件出错检测函数ferror	256	12.3 习题	265
11.7.3 清除文件出错标志和 结束标志函数clearerr	256	参考文献	267
11.8 习题	256	附录一 全国计算机等级考试二级 C 语言程序设计考试大纲 (2022 年版)	269
第 12 章 位运算	259	附录二 全国计算机等级考试二级 C 语言模拟题(一)	273
12.1 位运算符	259	附录三 全国计算机等级考试二级 C 语言模拟题(二)	287
12.1.1 按位与运算	259		
12.1.2 按位或运算	260		
12.1.3 按位异或运算	261		
12.1.4 按位求反运算	261		

第 1 章

引 言

人与计算机的交互是通过程序实现的，只有能够解决一定问题的程序才可以指挥计算机自动地进行工作，而程序又是通过程序设计语言开发的，其中 C 语言就是程序设计语言之一。本章主要介绍 C 语言的程序结构及特点，重点介绍在 Visual C++ 6.0 环境中实现 C 程序功能的步骤。

1.1 程序设计语言

程序是指人们使用编程语言开发、为解决一定问题且能够被计算机执行的指令代码。计算机程序设计语言是编程人员应遵守的、计算机可识别的程序代码规则，是人指挥计算机进行工作，与计算机进行交互的工具。

计算机程序设计语言一直在不断发展，纵观其历史，可以将其分为低级语言和高级语言两大类。

1.1.1 低级语言

低级语言又称为面向机器的语言，因 CPU 的不同而不同，可移植性差。使用低级语言可以编写出效率高的程序，但对程序设计人员的要求也很高。他们不仅要考虑解题思路，还要熟悉机器的内部结构，所以非专业人员很难掌握这类程序设计语言。

低级语言又分为机器语言和汇编语言。

1. 机器语言

机器语言是 CPU 可以直接识别的一组由 0 和 1 序列构成的指令代码。用机器语言编写程序，就是从所使用 CPU 的指令系统中挑选合适的指令，按照解决问题的算法组成一个指令序列。这种程序可以被机器直接理解并执行，速度很快，但因为它们不直观、难记、难写、不易查错、开发周期长、可移植性差，所以现在只有专业人员在编写对于执行速度有很高要求的程序时才采用机器语言。

2. 汇编语言

为了减轻编程者的劳动强度，人们使用一些帮助记忆的符号来代替机器语言中的 0、1 代码，使得编程效率和质量都有了很大的提高。由这些助记符组成指令系统的程序设计语言，称为符

号语言，也称为汇编语言。汇编语言指令与机器语言指令基本上是一一对应的，可移植性差。因为这些助记符不能被机器直接识别，所以用汇编语言编写的程序必须被汇编成机器语言后才能被机器理解。汇编之前的程序称为源程序，汇编之后的程序称为目标程序。可以使用连接程序将目标程序连接成可执行程序。可执行程序能够脱离语言环境独立运行。

1.1.2 高级语言

高级语言提供大量与人类语言相类似的控制结构，使程序设计者可以不关心机器的内部结构及工作原理，把主要的精力集中在解决问题的思路和方法上。这类摆脱了硬件束缚的程序设计语言的出现是计算机技术发展的里程碑，使得编程不再是少数专业人员的专利。由于高级语言不依赖具体的机器，因此用高级语言编写的程序可移植性较好。

根据编程机制的不同，高级语言又分为面向过程的程序设计语言和面向对象的程序设计语言。

1. 面向过程的程序设计语言

面向过程的程序设计语言由一个入口和一个出口构成，程序的每次执行都必须从这个入口开始，按照程序的结构执行到出口为止，这属于过程驱动的编程机制，由过程控制程序运行的流向。编程人员要以过程为中心来考虑应用程序的结构，执行哪一部分代码和按何种顺序执行代码都由程序本身控制。它允许将程序分解为多个函数，这使得同一个程序可以由多人分工开发，大大提高了编程效率，使人们能够开发出规模越来越大、功能越来越强的应用软件和系统软件。常用的面向过程的语言有 C、FORTRAN、Pascal 等。

2. 面向对象的程序设计语言

面向对象的程序设计语言将整个现实世界或者其中的一部分看作由不同种类的对象构成的，同一类型的对象既有相同点又有不同点。各种类型的对象之间通过发送消息进行联系，消息能够激发对象做出相应的反应，从而构成一个运动的整体，这属于事件驱动的编程机制，由事件控制着程序运行的流向。编程人员要以对象为中心来设计模块，代码不是按预定的顺序执行，而是在响应不同的事件时执行不同的代码。当前使用较多的面向对象的程序设计语言有 C++、C#、Java 等。

高级语言也不能被机器直接识别，也需要翻译后才能运行。高级语言的运行方式有解释和编译两种。所谓解释，是指边解释边执行，不生成目标代码，执行速度不快，源程序保密性不强，如 Visual Basic 属于解释方式。所谓编译，是将源程序使用语言本身提供的编译程序编译为目标程序，再使用连接程序与库文件连接成可执行程序，可执行程序能够脱离语言环境独立运行。本书中所介绍的 C 语言属于编译方式。

1.2 程序结构及其特点

C 语言是国际上广泛流行的面向过程的结构化程序设计高级语言。它是一种用途广泛、功能强大、使用灵活的编程语言，既可用于编写应用软件，又可用于编写系统软件。

1.2.1 程序结构

计算机程序设计语言有不同的语法规则和程序结构。C 语言程序结构如图 1-1 所示。

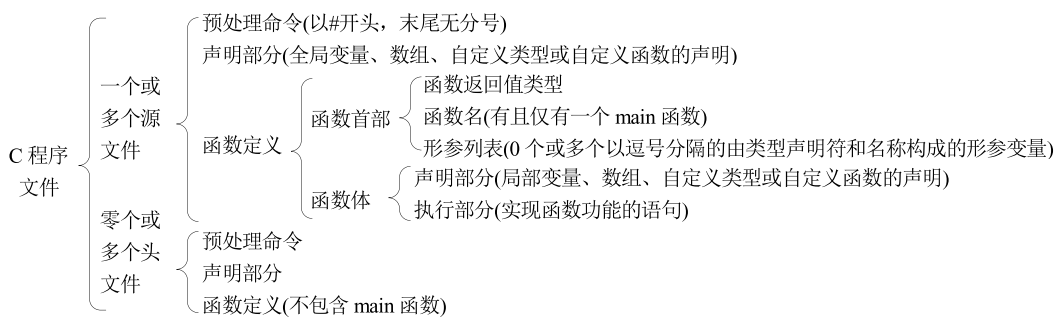


图 1-1 C 语言程序结构

1.2.2 程序结构的特点

通过分析图 1-1，可见 C 程序结构具有以下几个特点。

- (1) 一个 C 程序文件可以由一个或多个源文件(及零个或多个头文件)组成。
 - (2) 一个源文件或一个头文件可以依次包括 3 部分：预处理指令、声明部分和函数定义。
 - (3) 一个源文件可由一个或多个函数组成，但一个 C 程序有且仅有一个 main 函数，C 程序总是从 main 函数开始执行。
 - (4) 一个头文件可由零个或多个函数组成，但不能有 main 函数。
 - (5) 一个函数的定义包括以下两部分：
 - 函数首部：包括函数返回值类型、函数名、形参列表 3 部分。其中，形参列表由形参类型及形参名构成。
 - 函数体：包括声明部分和执行部分。其中，声明部分包括在本函数中所用到的局部变量或函数等的声明；执行部分由若干条语句组成，用于实现函数的功能。
 - (6) 一个声明或一条语句都必须以分号结尾，但预处理命令和函数首部的末尾不加分号。
- 为了更好地说明 C 程序结构的特点，下面以两个程序为例，演示组成 C 程序的基本结构和书写格式。

【例 1-1】在屏幕上输出信息“这是一个简单的 C 程序！”。

程序如下：

```
#include<stdio.h>           // include 为文件包含预处理命令(以#开头)
int main()                  // main 是主函数的函数名
{
    printf("这是一个简单的 C 程序! \n"); //直接调用系统定义的库函数 printf
    return 0;
}
```

程序运行结果如图 1-2 所示。



图 1-2 例 1-1 的运行结果

程序分析：**main** 是主函数的函数名，每个 C 程序都必须有且仅有一个 **main** 函数。在 **main** 函数之前的一行命令为预处理命令，这里的 **include** 称为文件包含预处理命令，其意义是把尖括号 `<>` 或引号 `" "` 内指定的文件包含到该程序中，成为该程序的一部分。被包含的文件通常是由系统提供的，其扩展名为 **.h**，因此也称为头文件。C 语言的头文件中包括了各个标准库函数的函数定义，因此，凡是在程序中调用库函数时，都必须包含该函数定义所在的头文件。**scanf** 和 **printf** 是标准输入/输出函数，其头文件为 **stdio.h**，由于在主函数前用 **include** 命令包含了 **stdio.h** 文件，因此可直接调用该文件。本例调用了输出函数 **printf**，把要输出的内容送到显示器显示。

【例 1-2】从键盘输入两个整数 **x** 和 **y**，求 **x** 与 **y** 的和，然后输出结果。

程序如下：

```
#include<stdio.h>           //扩展名为.h 的文件称为头文件
int main()
{
    int x,y,s;               //定义 3 个整型变量
    printf("input x:");      //显示第一个提示信息
    scanf("%d",&x);          //从键盘输入整数 x
    printf("input y:");      //显示第二个提示信息
    scanf("%d",&y);          //从键盘输入整数 y
    s=x+y;                   //求 x 与 y 的和，并把它赋给变量 s
    printf("sum of %d and %d is %d\n",x,y,s); //显示程序运行结果，即 s 的值
    return 0;
}
```

程序运行结果如图 1-3 所示。

程序分析：主函数体中又分为声明部分和执行部分两部分。声明部分是 C 程序结构中很重要的组成部分。C 语言规定，程序中所有用到的变量都必须先声明，后使用，否则会出错。例 1-1 中未使用任何变量，因此无声明部分。本例中使用了 3 个变

量 **x**、**y** 和 **s**，用来表示输入的自变量及求得的和。声明部分后的执行部分又称为执行语句部分，用于实现程序的功能。执行部分的第 1 行是输出语句，调用 **printf** 函数在显示器上给出提示字符串，提示操作人员输入自变量 **x** 的值。第 2 行是输入语句，调用 **scanf** 函数，接收键盘上输入的数并存入变量 **x** 中。第 3 行是输出语句，调用 **printf** 函数在显示器上给出提示字符串，提示操作人员输入自变量 **y** 的值。第 4 行是输入语句，调用 **scanf** 函数，接收键盘上输入的数并存入变量 **y** 中。第 5 行计算 **x** 与 **y** 的和，并把和赋给变量 **s**。第 6 行是用 **printf** 函数输出变量 **s** 的值。

运行本程序时，首先会在显示器上给出提示字符串 **input x:**，这是由执行部分的第 1 行完成的。用户在提示下从键盘上输入某个数，如 5，按 **Enter** 键，然后在显示器上给出提示字符串 **input y:**，这是由执行部分的第 3 行完成的。用户在提示下从键盘上输入某个数，如 7，按 **Enter** 键，接着在显示器上显示计算结果 12。

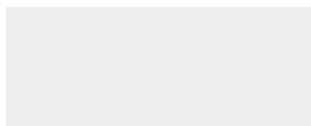


图 1-3 例 1-2 的运行结果

1.2.3 程序书写规则

从书写清晰，便于阅读、理解和维护的角度出发，在书写程序时应遵循以下规则。

(1) 一行可以写多个声明或语句，但为了清晰，一条声明或一条语句最好占一行。每条声明或语句都有明确的含义，能完成一定的任务。

(2) 用{}括起来的部分，通常表示程序的某一层结构。{}一般与该结构语句的第一个字母对齐，并单独占一行。

(3) 为了使程序便于阅读、易于调试，人们约定了锯齿形缩进的程序书写方式。将复合语句、函数体、循环体等语句用空格或 Tab 键向后缩进，使得程序错落有致，具有层次感。也就是说，低一层次的语句或声明比高一层次的语句或声明缩进若干空格。

(4) 标识符和关键字之间至少加一个空格以示分隔。若已有明显的分隔符，也可不再加空格。

(5) C 语言声明或语句中使用的都是西文字符(称半角字符)，所以在输入源程序时，应该将中文输入法关闭，避免输入全角字母和符号。全角字母和符号只有在字符串常量中才可以使用，而且字母是区分大小写的。

(6) 程序中适当地加上注释，以增强程序的可读性。

在编程时应力求遵循这些规则，以养成良好的编程习惯。

本书为了方便介绍语句、函数等的使用方法与语法格式，在命令格式中通常采用一些特殊的符号表示，如逗号加省略号、省略号等。这些符号不是命令的组成部分，在输入具体命令时，这些符号均不可作为语句中的成分输入计算机，它们只是命令格式的书面表示。具体含义如下。

- ... 表示同类的项可以重复多次。
- ... 表示省略了在当时叙述中不涉及的语句部分。

1.2.4 程序保留字

在 C 语言中使用的词汇分为标识符、关键字、运算符、分隔符、注释符和常量 6 类，除标识符外，其他均为保留字，有特定的作用，不能挪作他用。

1. 关键字

关键字是由 C 语言规定的具有特定意义的字符串。C 语言的关键字分为以下几类。

(1) 类型声明符：用于定义(或声明)变量、数组、自定义函数或自定义数据类型，如 int、float、double 等。

(2) 语句定义符：用于表示一条语句的功能，如 if、for、while 等。

(3) 预处理命令字：用于表示一个预处理命令，如例 1-1 和例 1-2 中用到的 include。

2. 运算符

C 语言中含有丰富的运算符。运算符与常量、变量、函数一起组成表达式，表示各种运算功能。运算符由一个或多个字符组成，如算术运算符+、-、*、/等。

3. 分隔符

在 C 语言中采用的分隔符有逗号和空格两种。逗号主要用在类型声明和函数参数表中分隔各个变量；空格多用作语句各单词之间的分隔符。在关键字、标识符之间必须要有一个或一个

以上的空格符作为分隔符，否则会出现语法错误。例如，若把“int a;”写成“inta;”，C 编译器会把 inta 当成一个标识符处理，其结果肯定出错。

4. 注释符

为了提高程序的可读性，通常在程序的适当位置加上必要的注释。C 语言的注释符有两种：一种是块注释，是以“/*”开头并以“*/”结尾的字符串；另一种是行注释，是以“//”开始到行尾的字符串。注释可出现在程序中的任何位置，注释主要用来解释语句或函数的功能，用来向用户提示或解释程序的意义，以便他人或开发者日后能够读懂程序。程序编译时，不对注释进行任何处理。在调试程序时可以对暂不使用的语句先用注释符括起来，使编译程序跳过处理，待调试结束后再去掉注释符。

5. 标识符

用来标识符号常量名、变量名、函数名、数组名、类型名、文件名等有效字符序列的符号，统称为标识符。除库函数的函数名由系统定义外，其余都由用户自己定义。

C 语言规定，标识符由字母(a~z, A~Z)、数字(0~9)、下画线(_)组成，并且第一个字符必须是字母或下画线，即标识符的命名规则是以字母或下画线开头，后面跟着字母、数字或下画线。

在使用标识符时还必须注意以下几点。

- (1) 标识符的长度受各种版本的 C 语言编译系统限制，同时也受具体机器的限制。
- (2) 标识符区分大小写，例如，b 和 B 是两个不同的标识符。
- (3) 标识符虽然可由程序员随意定义，但最好遵循见名知义的原则，以便于阅读和理解。

1.3 Visual C++ 6.0 的安装及使用

按照 C 程序结构的要求，所编写的解决某一具体问题的程序，需要有相应的编程环境来实现程序的功能。目前，实现 C 语言程序功能的编译系统有许多种，如 Visual C++ 6.0、Turbo C++ 3.0、GCC 等。本书以 Visual C++ 6.0(简称为 VC++ 6.0)作为开发平台。

1.3.1 Visual C++ 6.0 的安装

Visual C++ 6.0 是 Visual Studio 套装软件之一，它可以和 Visual Studio 一起安装，也可以单独安装。运行安装文件中的 setup.exe 文件，按照安装向导给出的提示，可以完成 Visual C++ 6.0 的安装。本书介绍的操作假定所使用的是单独安装的 Visual C++ 6.0。

1.3.2 Visual C++ 6.0 的启动

开机进入 Windows 后，可以通过多种方法启动 Visual C++ 6.0。

方法 1：双击 Windows 桌面上的 Visual C++ 6.0 的快捷方式图标(如果桌面上有)，这是最简单的启动方法。

方法 2：使用“开始”菜单中的程序命令。

如图 1-4 所示,单击 Windows 环境下的“开始”按钮,出现“开始”菜单,把鼠标指向“程序”菜单项,将出现“程序”子菜单。在“程序”子菜单中,把鼠标指向 Visual C++ 6.0 菜单项,出现 Visual C++ 6.0 子菜单,选择 Visual C++ 6.0 命令,即可进入 Visual C++ 6.0 编程环境,如图 1-5 所示。



图 1-4 启动 Visual C++ 6.0

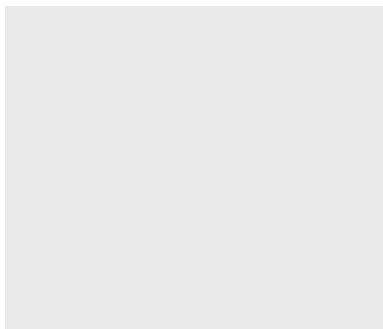


图 1-5 Visual C++ 6.0 编程环境

1.3.3 Visual C++ 6.0 上机过程

C 程序的运行过程如下:编辑源文件(.c),然后编译生成目标程序文件(.obj),再将其连接生成可执行文件(.exe),运行此文件得到程序结果。

1. 新建或打开一个源文件(.c 作为扩展名)

(1) 创建不包含工程的单个源文件。

进入图 1-5 所示的 Visual C++ 6.0 编程环境后,首先选择“文件”菜单下的“新建”命令,出现“新建”对话框,如图 1-6 所示,直接单击“新建”对话框的“文件”标签,打开“文件”选项卡,在列表框中选择 C++ Source File 选项,输入适当的位置及文件名,直接创建以.c 为扩展名的源文件。在此文件的编辑窗口中输入和编辑源文件,之后进行保存,编译时系统会提示创建工程工作区,如图 1-7 所示。

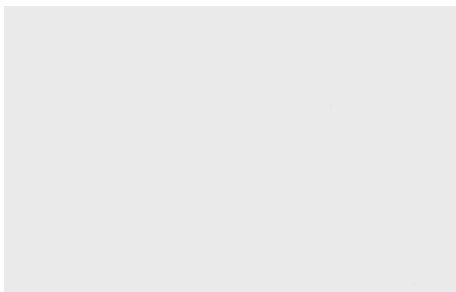


图 1-6 “新建”对话框



图 1-7 编译时系统提示新建工程工作区的对话框

(2) 打开一个已有的源文件

如果已经保存过源文件，希望打开它进行编辑，则具体方法如下。

① 在相应存储位置找到这个源文件。

② 双击此文件，自动进入 Visual C++ 6.0 集成环境，并打开该文件，此时，程序显示在编辑窗口中；也可以通过“文件”菜单下的“打开”命令，从中选择所需要的文件。

③ 如果将修改后的内容仍保存在原来的文件中，可以选择“文件”|“保存”命令。如果想保存到其他文件中，则选择“文件”|“另存为”命令。

2. 编译源文件，生成目标程序文件(.obj 作为扩展名)

选择“组建”|“编译”命令，对程序进行编译，出现系统提示，单击“是”按钮，在调试信息窗口中会显示错误和警告等提示信息。若有错误或者警告，双击出错信息，即可在源文件中定位错误。此时，需要返回第1步修改程序源代码，再编译，直到没有错误，才可进行下一步操作。

3. 连接目标程序，生成可执行文件(.exe 作为扩展名)

若编译没有错误，则可使用“组建”|“执行”命令，在调试信息窗口中会显示错误和警告等提示信息。若有错误或者警告，需要返回第1步修改程序源代码，再调试，直到没有错误，才可进行下一步操作。

4. 执行可执行文件，得到程序运行结果

若组建没有错误，则可使用“组建”|“执行”命令，直接进入结果窗口，显示最终结果。若结果不正确，则返回第1步修改程序源代码，再调试。

如果已完成程序的操作，不再对它进行其他处理，应选择“文件”|“关闭工作”命令，结束对该程序的操作，为编辑下一个程序做准备。

5. 创建和运行包含多个文件的程序的方法

上面介绍的是最简单的情况，即程序只包含一个源文件的情况。如果程序包含多个源文件，则需要创建一个工程文件，在这个工程文件中包含多个文件(包括源文件和头文件)。工程文件存放在工程工作区中，可由系统自动创建工程工作区。在编译时，系统会分别对工程文件中的每个文件进行编译，然后再将所得到的目标文件连接成一个整体，再与系统的有关资源进行连接，生成一个可执行文件，最后执行这个文件，得到结果。创建和运行包含多个文件的程序有如下两种方法。

方法一：先建工程，再建文件。

(1) 选择“文件”|“新建”命令，出现如图 1-8 所示的“新建”对话框，选择“工程”选项卡。在列表框中显示了可以在 Visual C++ 6.0 中创建的工程类型，选择 Win32 Console Application 工程类型，然后选择合适的存储位置，并输入工程名称，选中“创建新的工作空间”单选按钮，单击“确定”按钮，创建一个空工程。

(2) 选择“文件”|“新建”命令，出现如图 1-9 所示的“新建”对话框，选择“文件”选项卡。在列表框中选择 C++ Source File 选项，输入适当的文件名，可在该工程中创建以.c 为扩展名的源文件；选择 C/C++ Header File 选项，输入适当的文件名，可在该工程中创建以.h 为扩

展名的头文件。之后，在.c 文件的编辑窗口中输入和编辑源文件，或在.h 文件的编辑窗口中输入和编辑头文件，并保存。



图 1-8 “工程”选项卡

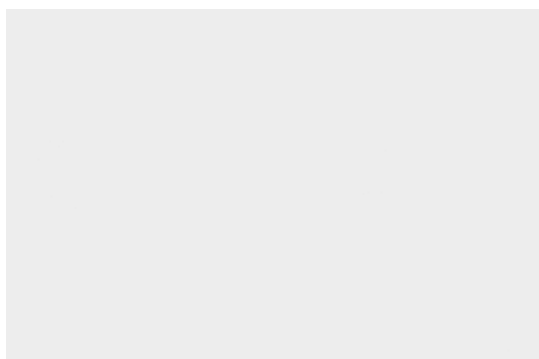


图 1-9 “文件”选项卡

(3) 编译和连接工程文件，生成可执行文件。

(4) 执行可执行文件。

方法二：先建文件，再建工程，将文件添加到工程中。

(1) 利用图 1-6 所示的对话框，分别编辑程序需要的各个文件(源文件以.c 为扩展名，头文件以.h 为扩展名)，并存放在指定的目录下。

(2) 利用图 1-8 所示的对话框创建一个工程。

(3) 将各个文件添加到工程中。方法是：选择“工程”|“增加到工程”子菜单下的“文件”命令，选择需要的各个文件，单击“确定”按钮，返回主窗口，选择工作区窗口下部的 fileview 选项卡，就可以看到工程中包含的所有源文件和头文件。

(4) 编译和连接工程文件，生成可执行文件。

(5) 执行可执行文件。

1.3.4 Visual C++ 6.0 的退出

退出 Visual C++ 6.0 很简单，只需要打开菜单栏中的“文件”菜单，并执行“退出”命令，或单击标题栏内最右边的“关闭”按钮，退出 Visual C++ 6.0 编程环境。如果当前程序已经修改过并且没有进行保存操作，退出时系统会显示一个对话框，询问用户是否进行保存。如果选择“是”，则保存后退出系统；如果选择“否”，则不进行保存退出系统；如果选择“取消”，则取消退出操作，返回 Visual C++ 6.0 编程环境。

1.4 习 题

一、选择题

- 如下关于 C 语言源程序的叙述中，错误的是()。
 - C 语言的源程序由函数构成
 - main 函数可以书写在自定义函数之后
 - 必须包含输入语句
 - 一行可以书写多条语句

2. 以下说法中正确的是()。
 - A. C 语言程序总是从第一个定义的函数开始执行
 - B. 在 C 语言程序中, 要调用的函数必须在 `main` 函数中定义
 - C. C 语言程序总是从 `main` 函数开始执行
 - D. C 语言程序中的 `main` 函数必须放在程序的开始部分
3. 在一个 C 语言程序中, `main` 函数的位置()。
 - A. 必须放在最开始
 - B. 必须在系统调用的库函数后面
 - C. 必须在最后
 - D. 可以任意
4. C 语言程序编译时, 程序中的注释部分将()。
 - A. 参加编译, 并会出现在目标程序中
 - B. 参加编译, 但不会出现在目标程序中
 - C. 不参加编译, 但会出现在目标程序中
 - D. 不参加编译, 也不会出现在目标程序中
5. 一个完整的可运行的 C 源程序()。
 - A. 至少要由一个主函数和(或)一个以上的辅助函数构成
 - B. 由一个且仅由一个主函数和零个以上(含零)的辅助函数构成
 - C. 至少要由一个主函数和一个以上的辅助函数构成
 - D. 至少由一个且只有一个主函数或多个辅助函数构成
6. 对于 C 语言源程序, 以下叙述中错误的是()。
 - A. 可以有空语句
 - B. 函数之间是平等的, 在一个函数内不能定义其他函数
 - C. 程序调试时如果没有提示错误, 就能得到正确结果
 - D. 注释可以出现在语句的前面
7. 以下叙述中正确的是()。
 - A. C 程序的每行只能写一条语句
 - B. 在对一个 C 程序进行编译的过程中, 可以发现注释中的拼写错误
 - C. C 语言本身没有输入、输出语句
 - D. 在 C 程序中, `main` 函数必须位于程序的最前面
8. 以下叙述中不正确的是()。
 - A. C 程序的执行总是从 `main` 函数开始
 - B. 一个 C 源程序必须包含一个 `main` 函数
 - C. C 语言程序的基本组成单位是函数
 - D. 在编译 C 源程序时, 可发现注释中的拼写错误
9. 下面关于 C 语言用户标识符的描述中, 正确的是()。
 - A. 不能区分大小写
 - B. 用户标识符不能描述常量
 - C. 类型名也是用户标识符
 - D. 用户标识符可以作为变量名

二、填空题

1. 一条 C 语句中至少包含一个_____。
2. C 语言标识符由_____、_____和下画线构成。