



普通高等教育“十二五”规划教材

# C 语言程序设计

姜雪◎ 主编



科学出版社

普通高等教育“十二五”规划教材

# C 语言程序设计

姜 雪 主编  
张春芳 刘立君 张 宇 副主编

科学出版社

北 京

## 内 容 简 介

本书全面介绍了 C 语言的基本语法知识及运用 C 语言进行程序设计的相关内容,在内容编排上,力求通俗易懂,循序渐进,重点突出。为了使读者更好地掌握各章节知识,提高逻辑分析和程序设计能力,每章末均配有精选的习题。

本书中的程序均按照模块化程序设计思想进行编写,并辅以必要的注释,便于读者对程序的分析、理解和自学。书中涉及的所有程序已在 Microsoft Visual C++6.0 集成开发环境中进行了调试和运行,程序算法采用 N-S 图描述。

本书可作为普通高等院校各专业 C 语言程序设计的教学用书,也可作为 C 语言的初学者和相关培训机构的参考书或培训教材。

---

### 图书在版编目(CIP)数据

C 语言程序设计/姜雪主编. —北京:科学出版社,2015

(普通高等教育“十二五”规划教材)

ISBN 978-7-03-043511-8

I. ①C… II. ①姜… III. ①C 语言—程序设计—高等学校—教材  
IV. ①TP312

中国版本图书馆 CIP 数据核字(2015)第 034764 号

---

责任编辑:宋丽 张斌 / 责任校对:马英菊

责任印制:吕春珉 / 封面设计:多边设计

**科学出版社** 出版

北京东黄城根北街 16 号

邮政编码:100717

<http://www.sciencep.com>

**北京源海印刷有限公司** 印刷

科学出版社发行 各地新华书店经销

\*

2015 年 2 月第 一 版 开本:787×1092 1/16

2015 年 2 月第一次印刷 印张:16 3/4

字数:400 000

定价:34.00 元

(如有印装质量问题,我社负责调换<路通>)

销售部电话 010-62134988 编辑部电话 010-62135741

**版权所有,侵权必究**

举报电话:010-64030229; 010-64034315; 13501151303

# 前 言



C 语言是一种结构化程序设计语言，其代码简洁、高效，功能强大。C 语言既可用于编写应用程序，也可用于编写系统程序。从实用性、易用性和学习的难易程度等角度看，C 语言是不可多得的计算机高级语言，而 C 语言程序设计也是大部分高校计算机及相关专业的必修课程。

本书共分 11 章。第 1 章主要介绍了 C 语言的基本知识和在 Microsoft Visual C++ 6.0 集成开发环境下的上机调试过程。第 2 章主要介绍了数据类型、运算符和表达式及基本输入/输出函数等。第 3~5 章分别介绍了顺序结构、选择结构、循环结构控制语句及基本程序设计方法，可以解决比较复杂的实际问题。第 6 章主要介绍了一维数组、二维数组、字符数组的定义和使用，以及常用的字符串处理函数。第 7 章主要介绍了函数的定义、调用和参数的使用，以及变量的作用域等。第 8 章主要介绍了编译预处理命令。第 9 章主要介绍了指针的定义和使用，这是 C 语言学习的重点与难点，也是体现 C 语言“高级”能力的知识点。第 10 章主要介绍了结构体、共用体和枚举类型的定义和使用，以及单向链表的相关内容等。第 11 章主要介绍了文件操作、读/写函数及文件的定位和出错检验等。

本书是根据编者多年的教学经验编写而成的，由浅入深，循序渐进，将理论与实践有机结合，将知识传授与能力培养融为一体。通过本书的学习，读者既能快速掌握 C 语言的基础知识，又能很快学会 C 语言的编程技巧，提高解决实际问题的能力。

本书由姜雪担任主编，张春芳、刘立君、张宇担任副主编，张宇担任主审。第 1 章、第 6 章、第 7 章由刘立君编写，第 2 章、第 3 章、第 5 章由姜雪编写，第 4 章、附录由张宇编写，第 8~11 章由张春芳编写。全书由姜雪、张宇统稿。

由于编者水平有限，书中难免存在缺点和不足之处，恳请有关专家和读者批评指正。

# 目 录



## 前言

|                                       |    |
|---------------------------------------|----|
| 第 1 章 C 语言概述                          | 1  |
| 1.1 C 语言的发展及特点                        | 1  |
| 1.1.1 C 语言的发展                         | 1  |
| 1.1.2 C 语言的特点                         | 1  |
| 1.2 C 语言的标识符与关键字                      | 2  |
| 1.2.1 字符集                             | 2  |
| 1.2.2 标识符                             | 2  |
| 1.2.3 关键字                             | 3  |
| 1.3 C 语言程序概述                          | 3  |
| 1.3.1 C 语言程序的基本组成                     | 3  |
| 1.3.2 C 语言程序的书写风格                     | 5  |
| 1.4 C 语言程序上机调试                        | 6  |
| 1.4.1 Microsoft Visual C++ 6.0 集成开发环境 | 6  |
| 1.4.2 C 语言程序的上机调试步骤                   | 6  |
| 习题                                    | 9  |
| 第 2 章 数据类型、运算符与表达式                    | 10 |
| 2.1 数据类型                              | 10 |
| 2.2 常量                                | 11 |
| 2.2.1 整型常量                            | 11 |
| 2.2.2 实型常量                            | 12 |
| 2.2.3 字符常量                            | 12 |
| 2.2.4 字符串常量                           | 14 |
| 2.2.5 符号常量                            | 14 |
| 2.3 变量                                | 15 |
| 2.3.1 变量的定义和初始化                       | 15 |
| 2.3.2 整型变量                            | 17 |
| 2.3.3 实型变量                            | 17 |

|                                |    |
|--------------------------------|----|
| 2.3.4 字符变量 .....               | 18 |
| 2.4 基本输入/输出函数 .....            | 19 |
| 2.4.1 字符输出函数——putchar( ) ..... | 19 |
| 2.4.2 字符输入函数——getchar( ) ..... | 20 |
| 2.4.3 格式输出函数——printf( ) .....  | 21 |
| 2.4.4 格式输入函数——scanf( ) .....   | 25 |
| 2.5 运算符和表达式 .....              | 29 |
| 2.5.1 算术运算符和算术表达式 .....        | 30 |
| 2.5.2 强制类型转换运算符 .....          | 32 |
| 2.5.3 赋值运算符和赋值表达式 .....        | 33 |
| 2.5.4 自加、自减运算符及其表达式 .....      | 35 |
| 2.5.5 关系运算符和关系表达式 .....        | 36 |
| 2.5.6 逻辑运算符和逻辑表达式 .....        | 37 |
| 2.5.7 条件运算符和条件表达式 .....        | 39 |
| 2.5.8 逗号运算符和逗号表达式 .....        | 41 |
| 2.5.9 求字节数运算符 .....            | 41 |
| 2.5.10 位运算符和位运算 .....          | 42 |
| 习题 .....                       | 44 |
| 第3章 顺序结构程序设计 .....             | 48 |
| 3.1 程序设计基础 .....               | 48 |
| 3.2 C语言的基本语句 .....             | 50 |
| 3.3 顺序结构程序设计举例 .....           | 52 |
| 习题 .....                       | 53 |
| 第4章 选择结构程序设计 .....             | 54 |
| 4.1 if语句 .....                 | 54 |
| 4.2 switch语句 .....             | 62 |
| 4.3 选择结构程序设计举例 .....           | 64 |
| 习题 .....                       | 67 |
| 第5章 循环结构程序设计 .....             | 72 |
| 5.1 while语句 .....              | 72 |
| 5.2 do···while语句 .....         | 75 |
| 5.3 for语句 .....                | 78 |
| 5.4 转向语句 .....                 | 83 |

|                         |            |
|-------------------------|------------|
| 5.4.1 break 语句 .....    | 83         |
| 5.4.2 continue 语句 ..... | 85         |
| 5.4.3 goto 语句 .....     | 86         |
| 5.5 多重循环 .....          | 87         |
| 5.6 循环结构程序设计举例 .....    | 89         |
| 习题 .....                | 92         |
| <b>第 6 章 数组</b> .....   | <b>99</b>  |
| 6.1 一维数组 .....          | 99         |
| 6.1.1 一维数组的定义 .....     | 99         |
| 6.1.2 一维数组元素的引用 .....   | 100        |
| 6.1.3 一维数组的初始化 .....    | 101        |
| 6.1.4 一维数组程序应用举例 .....  | 102        |
| 6.2 二维数组 .....          | 107        |
| 6.2.1 二维数组的定义 .....     | 107        |
| 6.2.2 二维数组元素的引用 .....   | 108        |
| 6.2.3 二维数组的初始化 .....    | 108        |
| 6.2.4 二维数组程序应用举例 .....  | 109        |
| 6.3 字符数组 .....          | 113        |
| 6.3.1 字符数组的定义 .....     | 113        |
| 6.3.2 字符数组元素的引用 .....   | 113        |
| 6.3.3 字符数组的初始化 .....    | 113        |
| 6.3.4 字符数组的输入/输出 .....  | 114        |
| 6.3.5 常用的字符串处理函数 .....  | 116        |
| 习题 .....                | 120        |
| <b>第 7 章 函数</b> .....   | <b>123</b> |
| 7.1 函数概述 .....          | 123        |
| 7.1.1 函数的分类 .....       | 123        |
| 7.1.2 函数的定义 .....       | 124        |
| 7.2 函数的调用和声明 .....      | 125        |
| 7.2.1 函数的调用 .....       | 125        |
| 7.2.2 函数的声明 .....       | 126        |
| 7.3 函数的参数传递方式和返回值 ..... | 127        |
| 7.3.1 函数的参数传递方式 .....   | 127        |
| 7.3.2 函数的返回值 .....      | 130        |

|                          |            |
|--------------------------|------------|
| 7.4 函数的嵌套调用和递归调用 .....   | 132        |
| 7.4.1 函数的嵌套调用 .....      | 132        |
| 7.4.2 函数的递归调用 .....      | 133        |
| 7.5 变量的作用域与存储 .....      | 135        |
| 7.5.1 变量的作用域 .....       | 135        |
| 7.5.2 变量的存储方式 .....      | 138        |
| 7.5.3 变量的存储类别 .....      | 139        |
| 7.6 内部函数和外部函数 .....      | 142        |
| 7.6.1 内部函数 .....         | 142        |
| 7.6.2 外部函数 .....         | 143        |
| 习题 .....                 | 143        |
| <b>第8章 编译预处理</b> .....   | <b>146</b> |
| 8.1 文件包含 .....           | 146        |
| 8.2 宏定义 .....            | 148        |
| 8.2.1 不带参数的宏定义 .....     | 148        |
| 8.2.2 带参数的宏定义 .....      | 151        |
| 8.3 条件编译 .....           | 152        |
| 习题 .....                 | 154        |
| <b>第9章 指针</b> .....      | <b>158</b> |
| 9.1 指针的概念 .....          | 158        |
| 9.1.1 地址及指针 .....        | 158        |
| 9.1.2 变量的指针 .....        | 158        |
| 9.2 指针变量 .....           | 159        |
| 9.2.1 指针变量的定义及初始化 .....  | 159        |
| 9.2.2 指针变量的引用 .....      | 162        |
| 9.3 指针与一维数组 .....        | 164        |
| 9.3.1 数组的指针和指针变量 .....   | 164        |
| 9.3.2 指针运算 .....         | 165        |
| 9.3.3 通过指针引用数组的元素 .....  | 166        |
| 9.4 字符指针与字符串 .....       | 170        |
| 9.4.1 字符指针变量 .....       | 170        |
| 9.4.2 字符指针与字符数组的区别 ..... | 171        |
| 9.5 指针与二维数组 .....        | 171        |
| 9.5.1 二维数组的指针 .....      | 171        |

|               |                |            |
|---------------|----------------|------------|
| 9.5.2         | 二维数组元素的表示方法    | 172        |
| 9.5.3         | 行指针变量          | 174        |
| 9.6           | 指针数组           | 175        |
| 9.7           | 指针作为函数的参数      | 179        |
| 9.8           | 多级指针           | 181        |
|               | 习题             | 183        |
| <b>第 10 章</b> | <b>结构体与共用体</b> | <b>188</b> |
| 10.1          | 结构体类型与结构体类型变量  | 188        |
| 10.1.1        | 结构体类型的定义       | 188        |
| 10.1.2        | 结构体类型变量        | 189        |
| 10.1.3        | 结构体类型变量的引用     | 192        |
| 10.2          | 结构体类型数组        | 194        |
| 10.2.1        | 结构体类型数组的定义及初始化 | 194        |
| 10.2.2        | 结构体类型数组元素的引用   | 195        |
| 10.3          | 指向结构体类型数据的指针变量 | 197        |
| 10.3.1        | 指向结构体类型变量的指针变量 | 197        |
| 10.3.2        | 指向结构体类型数组的指针变量 | 199        |
| 10.4          | 单向链表           | 200        |
| 10.4.1        | 单向链表的数据结构      | 201        |
| 10.4.2        | 动态分配和释放空间函数    | 201        |
| 10.4.3        | 单向链表的基本操作      | 202        |
| 10.5          | 共用体类型          | 213        |
| 10.5.1        | 共用体类型的定义       | 213        |
| 10.5.2        | 共用体类型变量的定义     | 213        |
| 10.5.3        | 共用体类型变量的引用     | 214        |
| 10.5.4        | 共用体类型变量的赋值     | 215        |
| 10.6          | 枚举类型           | 217        |
| 10.7          | 为已有类型定义起别名     | 219        |
|               | 习题             | 220        |
| <b>第 11 章</b> | <b>文件</b>      | <b>221</b> |
| 11.1          | 文件概述           | 221        |
| 11.1.1        | 文件系统概述         | 221        |
| 11.1.2        | 文件类型指针         | 222        |
| 11.2          | 文件的打开与关闭       | 222        |

|                             |     |
|-----------------------------|-----|
| 11.2.1 文件的打开 .....          | 223 |
| 11.2.2 文件的关闭 .....          | 224 |
| 11.3 文件的读/写函数 .....         | 225 |
| 11.3.1 字符读/写函数 .....        | 225 |
| 11.3.2 字符串读/写函数 .....       | 228 |
| 11.3.3 格式化读/写函数 .....       | 229 |
| 11.3.4 数据块读/写函数 .....       | 232 |
| 11.4 文件的定位 .....            | 233 |
| 11.5 文件的出错检验 .....          | 234 |
| 习题 .....                    | 235 |
| 附录 .....                    | 239 |
| 附录 1 常用字符与 ASCII 码对照表 ..... | 239 |
| 附录 2 C 语言常用库函数 .....        | 241 |
| 习题参考答案 .....                | 246 |
| 参考文献 .....                  | 258 |

# 第 1 章 C 语言概述

## 1.1 C 语言的发展及特点

C 语言是一种应用广泛的程序设计语言，既可以用来编写系统软件，又可以编写应用软件。目前，C 语言为大学程序设计课程的首选语言。

### 1.1.1 C 语言的发展

C 语言的起源可以追溯到 ALGOL 60 语言。1963 年，剑桥大学在 ALGOL 60 语言的基础上推出了 CPL (Combined Programming Language)。1967 年，剑桥大学的 Martin Richards 对 CPL 进行了简化，推出了 BCPL (Basic Combined Programming Language)。

1970 年，美国贝尔实验室的 Ken Thompson 对 BCPL 进行了修改，设计出了 B 语言，并且他用 B 语言写了第一个 UNIX 操作系统，在 DEC PDP-7 计算机上实现。1973 年，美国贝尔实验室的 D.M.Ritchie 对 B 语言进行了进一步简化和修改，最终设计出了 C 语言，C 语言首次在 UNIX 操作系统的 DEC PDP-11 计算机上使用。

为了推广 UNIX 操作系统，1977 年 D.M.Ritchie 发表了不依赖于具体机器系统的 C 语言编译文本《可移植的 C 语言编译程序》。1978 年，B.W.Kernighan 和 D.M.Ritchie 出版了著名的 *The C Programming Language*，从而使 C 语言成为目前世界上应用最广泛的高级程序设计语言。

1988 年，随着微型计算机的日益普及，出现了许多 C 语言版本。由于没有统一的标准，这些 C 语言之间出现了一些不一致的地方。为了改变这种情况，美国国家标准化协会 (ANSI) 为 C 语言制定了一套 ANSI 标准，这套标准成为现行的 C 语言标准。此后的 Turbo C 和 Microsoft Visual C++ (简称 Visual C++) 编译系统都是按照标准的 C 语言编写的。本书将 Visual C++ 6.0 作为软件开发环境。

C 语言发展迅速，而且成为受欢迎的程序设计语言之一，主要是因为其具有强大的功能。许多著名的系统软件都是用 C 语言编写的，用 C 语言编写在对操作系统及需要对硬件进行操作的场合，C 语言明显优于其他高级语言。

### 1.1.2 C 语言的特点

(1) 简洁紧凑、灵活方便。C 语言一共只有 32 个关键字和 9 种控制语句，程序书写自由，主要用小写字母表示。

(2) 运算符丰富。C 语言的运算符很多，共有 40 多个运算符。C 语言把括号、赋

值、强制类型转换等作为运算符处理，从而使 C 语言的运算类型极其丰富，表达式类型多样化。灵活使用各种运算符可以实现在其他高级语言中难以实现的运算。

(3) 数据结构丰富。C 语言的数据类型有整型、实型、字符型、数组类型、指针型等，可以实现各种复杂的数据类型的运算。C 语言引入了指针概念，使程序效率更高。另外，C 语言具有强大的图形功能，支持多种显示器和驱动器，而且计算功能、逻辑判断功能强大。

(4) 结构化语言。C 语言是一种结构化语言。结构化语言的显著特点是代码及数据的分隔化，即程序的各个部分除了必要的信息交流外，彼此独立。这种结构化方式可使程序层次清晰，便于使用、维护及调试。C 语言是以函数形式提供给用户的，这些函数方便调用，并具有多种循环语句、条件语句控制程序流向，从而使程序完全结构化。

(5) 允许直接访问物理地址，可以直接对硬件进行操作。C 语言既具有高级语言的特点，又具有低级语言的许多功能，能够像汇编语言一样对位、字节和地址进行操作，而这三者是计算机的基本工作单元，可以用来编写系统软件。

(6) 生成代码质量高，程序执行效率高。C 语言程序一般只比汇编程序生成的目标代码效率低 10%~20%。

(7) 适用范围大，可移植性好。C 语言有一个突出的优点，即适用于多种操作系统，如 DOS、UNIX 等，也适用于多种机型。

## 1.2 C 语言的标识符与关键字

### 1.2.1 字符集

字符是组成程序设计语言的最基本的元素。C 语言字符集由字母、数字、空格、标点和特殊字符组成。在字符常量、字符串常量和注释中还可以使用汉字或其他可表示的图形符号。

(1) 字母：小写字母 a~z 共 26 个，大写字母 A~Z 共 26 个。

(2) 数字：0~9 共 10 个。

(3) 空白符：空格符、制表符 (Tab 键)、换行符等统称为空白符。空白符只在字符常量和字符串常量中起作用。空白符在其他地方出现时，只起间隔作用，编译程序对其忽略不计。

(4) 标点和特殊字符：如加号 (+)、减号 (-)、分号 (;)、逗号 (,) 等。

### 1.2.2 标识符

标识符用来标识源程序中的各个对象的名称，这些对象可以是语句、数据类型、函数、变量、常量、数组等。C 语言中标识符要遵循以下规则。

(1) 一个标识符只能由英文字母、数字和下划线 3 种字符组成，并且英文的大写字

母和小写字母被认为是不同字符。例如，对于 `sec` 和 `SEC` 这两个标识符来说，C 语言会认为这是两个完全不同的标识符。

(2) 第一个字符必须是字母或下划线，通常以下划线开头的标识符是编译系统专用的，所以在编写 C 语言程序时，最好不要使用以下划线开头的标识符。但是下划线可以用在第一个字符以后的任何位置。例如，`a`、`x`、`_x3`、`BOOK_1`、`sum5` 等标识符是合法的，而 `3s`、`book*`、`3-s` 等标识符是非法的。

(3) 不能使用系统的关键字作为用户标识符，因为系统关键字已有确定含义。标识符虽然可由用户随意定义，但由于标识符是用于标识某个量的符号，因此 C 语言程序中的标识符名称应简洁明了、含义清晰，做到“顾名思义”，这样便于程序的阅读和维护。

### 1.2.3 关键字

关键字是由 C 语言规定的具有特定意义的字符串，通常也称为保留字。用户定义的标识符不应与关键字相同。C 语言规定的关键字共有 32 个，都用小写字母书写。C 语言中的关键字如表 1-1 所示。

表 1-1 C 语言中的关键字

| 类 型     | 成 员                                                                                                                                                                                                                                                       |
|---------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 控制类型关键字 | <code>if</code> , <code>else</code> , <code>switch</code> , <code>case</code> , <code>break</code> , <code>default</code> , <code>while</code> , <code>do</code> , <code>for</code> , <code>goto</code> , <code>return</code> , <code>continue</code>     |
| 数据类型关键字 | <code>void</code> , <code>char</code> , <code>int</code> , <code>float</code> , <code>short</code> , <code>long</code> , <code>double</code> , <code>signed</code> , <code>unsigned</code> , <code>struct</code> , <code>union</code> , <code>enum</code> |
| 存储类型关键字 | <code>static</code> , <code>extern</code> , <code>auto</code> , <code>register</code>                                                                                                                                                                     |
| 其他关键字   | <code>volatile</code> , <code>const</code> , <code>typedef</code> , <code>sizeof</code>                                                                                                                                                                   |

## 1.3 C 语言程序概述

### 1.3.1 C 语言程序的基本组成

C 语言程序是由函数组成的。下面以一个最简单的程序为例说明 C 语言程序的基本组成。

**【例 1-1】**编写程序，在屏幕上显示“Hello, world!”。

```
/* 第一个C语言程序,在屏幕上显示"Hello, world!" */
#include<stdio.h>           //包含头文件 stdio.h
void main()                 // 主函数
{
    printf("Hello, World!\n"); // 输出字符串
}
```

程序运行结果:

```
Hello,World!
```

说明:

第 1 行包含标准库文件,include 称为文件包含命令,扩展名为.h 的文件称为头文件。

第 2 行定义名为 main 的函数,main() 函数不接受参数值,其语句都被括在花括号中。void 说明 main() 函数无返回值类型。

第 3 行开始 main() 函数,“{” 是 main() 函数的开始标志。

第 4 行显示 “Hello, world!”, main() 函数调用库函数 printf() 以输出字符串,语句以分号结束。

第 5 行结束 main() 函数,“}” 是 main() 函数的结束标志。

注意: 花括号必须成对出现。

**【例 1-2】**编写程序,求两个数的最小值。

```
#include<stdio.h>
int min(int x,int y)          //最小值函数
{
    int z;                    //声明整型变量 z
    if(x<y)                   //求变量 x,y 的最小值
        z=x;
    else
        z=y;
    return z;                 //返回变量 z 的值
}

void main()
{
    int a,b,c;                //声明整型变量 a,b,c
    scanf("%d %d",&a,&b);      //输入变量 a,b 的值
    c=min(a,b);               //调用 min() 函数
    printf("min=%-3d\n",c);    //输出变量 c 的值
}
```

程序运行时输入:

```
34 85
```

程序运行结果:

```
min=34
```

说明:

本程序由 `main()` 函数和 `min()` 函数组成。程序由 `main()` 函数开始, 执行到 `c=min(a,b)` 时调用 `min()` 函数, `min()` 函数调用结束后, 返回 `main()` 函数。

`min()` 函数中第 1 行是函数首部, 依次说明函数类型、函数名、参数。第 2~9 行, 即花括号内的部分是 `min()` 函数的函数体, 其中第 3 行是函数体的声明部分, 声明变量 `z`, 第 4~7 行是一个选择结构, 把变量 `x`、`y` 中的最小值赋值给变量 `z`, 第 8 行返回变量 `z` 的值。

`main()` 函数中第 1 行是函数首部。第 2~7 行是函数体, 其中第 3 行是函数体的声明部分, 声明整型变量 `a`、`b`、`c`; 第 4~6 行是执行部分, 其中第 4 行是输入函数, 输入变量 `a`、`b` 的值, 第 5 行调用 `min()` 函数, 第 6 行输出变量 `c` 的值。

通过上述两个例子, 可以把 C 语言程序的结构特点概括为以下几点。

(1) C 语言程序可由一个或多个函数组成。一个源程序无论由多少个函数组成, 有且只能有一个 `main()` 函数, 即主函数。

(2) 无论 `main()` 函数写在程序中什么位置, 程序总是从 `main()` 函数开始执行。

(3) 一个函数由函数首部和函数体两部分组成。函数首部即函数的第 1 行, 包括函数类型、函数名、参数类型和参数名。例如:

|                  |                  |                |                  |                 |                  |                 |
|------------------|------------------|----------------|------------------|-----------------|------------------|-----------------|
| <code>int</code> | <code>min</code> | <code>(</code> | <code>int</code> | <code>x,</code> | <code>int</code> | <code>y)</code> |
| 函数类型             | 函数名              |                | 参数类型             | 参数名             | 参数类型             | 参数名             |

函数可以没有参数, 但是后面的一对括号不能省略, 这是格式的规定。

函数体是函数中用一对花括号括起来的部分。如果函数体内有多个花括号, 最外层的花括号内的部分是函数体的范围。函数体一般包括声明部分和执行部分两部分。

```
{  
    声明部分: 定义本函数所使用的变量  
    执行部分: 由若干条语句组成命令序列 (可以在其中调用其他函数)  
}
```

(4) C 语言本身不提供输入/输出语句, 输入/输出操作通过调用库函数 (`scanf/printf`) 完成。

(5) 程序中只要用到库函数, 都要有 `#include` 预处理命令, 预处理命令通常放在函数最前面。

(6) 位于 `/* */` 中和 `//` 后面的内容为注释, 用来对程序进行说明。注释在编译时会被自动忽略, 程序运行时该部分不执行。

### 1.3.2 C 语言程序的书写风格

从书写清晰, 便于阅读、理解和维护的角度出发, 用 C 语言编写程序时应遵循以下规则。

(1) 一个语句可以占多行，一行可以写多个语句，每一个注释和每一个语句都必须以分号结尾。但预处理命令、函数首部和花括号“{”后面不能加分号。

(2) 标识符与关键字之间必须至少加一个空格以示间隔。若已有明显的间隔符，也可不再加空格来间隔。

(3) 用花括号括起来的部分通常表示程序的某一层结构。花括号一般与该结构语句的第一个字母对齐，并单独占一行。

(4) 低一层次的语句或注释可在高一层次的语句或注释缩进若干空格后书写，以使程序更加清晰，增强程序的可读性。

(5) 函数和函数之间加空行，以便清晰地分辨出程序中函数的数量。

(6) 为了增强程序的可读性，对语句和函数应加上适当的注释。

## 1.4 C 语言程序上机调试

### 1.4.1 Microsoft Visual C++ 6.0 集成开发环境

Microsoft Visual C++ 6.0（简称 VC 6.0）是 Microsoft 公司推出的目前使用极为广泛的基于 Windows 平台的可视化编程环境。由于其功能强大、灵活性好、完全可扩展及具有强有力的 Internet 支持，VC 6.0 成为目前最为流行的 C++ 语言集成开发环境。VC 6.0 集成环境中集合了程序编辑器、编译器、连接程序和执行程序。

安装 VC 6.0 后，若桌面上已自动生成了 VC 6.0 快捷方式图标，则双击快捷方式图标即可启动 VC 6.0。若桌面没有快捷方式图标，则单击“开始”按钮，在弹出的菜单中选择“Microsoft Visual C++ 6.0”选项，即可启动 VC 6.0，如图 1-1 所示。

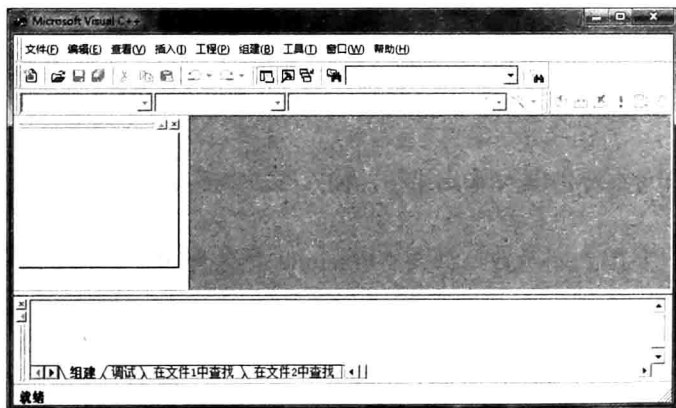


图 1-1 VC 6.0 窗口

### 1.4.2 C 语言程序的上机调试步骤

在 VC 6.0 环境下调试 C 语言程序需要 4 个步骤，即程序编辑、编译、连接和运行。

1. 程序编辑

选择“文件→新建”选项，打开如图 1-2 所示的“新建”对话框。选择“文件”选项卡，从中选择“C++ Source File”文件类型，单击“确定”按钮，打开如图 1-3 所示的程序编辑窗口。在编辑区输入程序代码，编辑修改完成后保存。VC 6.0 源程序默认的扩展名为.cpp。

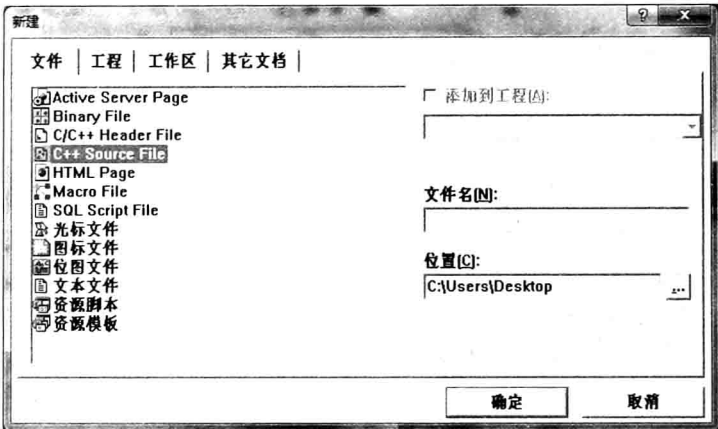


图 1-2 “新建”对话框

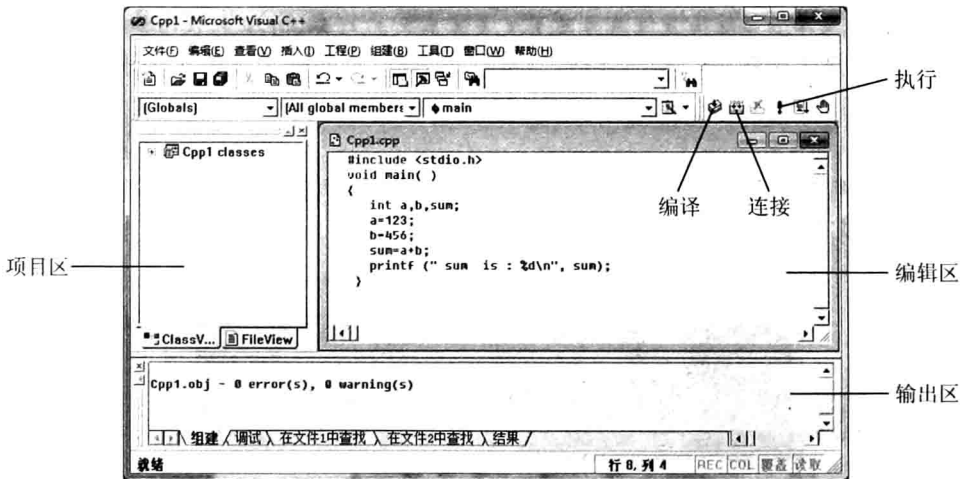


图 1-3 程序编辑窗口

2. 编译

将 VC 6.0 源程序 (.cpp) 进行编译后，生成目标程序 (.obj)。编译过程中对程序进行语法等项目的检查，并将编译结果显示在输出区。如图 1-3 所示，单击“编译”按钮