

教育科学“十五”国家规划课题研究成果

# C 程序设计

王柏盛            主 编  
李万庆 贺洪江   副主编

高等教育出版社

## 内 容 提 要

本书全面介绍了 Turbo C 语言的基本概念,常量、变量、运算符和表达式,程序控制语句,函数、指针、结构、联合、枚举和定义类型、编译预处理命令、文件、字符屏幕和图形函数以及实用编程技术等内容。全书共分为 10 章。每章附有习题和实验,并精选了一部分全国计算机等级考试(二级 C 语言程序设计)的练习题,通过大量实例介绍 C 程序设计的思想、方法和技巧。

作者根据多年教学和科研积累的丰富经验,吸取当前一些 C 语言教材中的优点,增加了实用编程技术方面的内容,力求使本书体系合理、结构严谨、概念清晰、例题丰富、通俗易懂。

本书可作为高等院校程序设计课程的教材,也可供自学者使用或作为教师教学参考书。

与本书配套的辅助教材《C 程序设计习题题解》也同时出版。

# 前 言

C 语言是在国内外得到广泛使用的结构化程序设计语言,许多大型开发工具都以 C 语言作为基础语言。C 语言功能丰富、表达力强,使用方便灵活,目标程序效率高,可移植性好,既有结构化高级程序设计语言的优点,又有低级语言的绝大部分功能,它可以取代汇编语言,用来编写大型系统软件、工具软件和控制软件。

现在,C 语言已不仅为计算机专业工作者所使用,而且为广大工程技术人员所喜爱,许多人已经用它编写出很多优秀的应用软件。高等院校在非计算机专业普遍开设了 C 语言程序设计课程,大部分学生都选择 C 语言参加全国计算机等级考试。

本书全面介绍了 Turbo C 语言的基本概念,常量、变量、运算符和表达式,程序控制语句,函数,指针,结构、联合、枚举和定义类型,编译预处理命令,文件,字符屏幕和图形函数以及实用编程技术等内容。全书共 10 章,每章均附有习题和实验,并精选了一部分全国计算机等级考试(二级 C 语言程序设计)的练习题,通过大量实例介绍了 C 程序设计的思想、方法和技巧。

作者根据多年教学和科研积累的丰富经验,吸取当前一些 C 语言教材中的优点,大篇幅增加了字符屏幕、图形函数和实用编程技术方面的内容,力求使本书集教材、参考资料为一体,具有较强的实用性。本书的主要特点可以概括为以下几个方面:

(1) 体系合理、结构严谨、概念清晰、例题丰富,在章节安排上,更符合结构化程序设计语言的特点。

(2) 在程序设计技巧上有所创新,充分运用了 C 语言各种运算符和表达式的功能。各章设计了很多例程序并提供多种解法,以加强读者阅读能力、编程能力、调试能力和创新能力的培养。

(3) 增加了字符屏幕、图形函数及实用编程技术等章节,使本书在深度和广度上都得到了加强。

(4) 指出了 Turbo C 在表达式中存在的严重不一致性。

讲解本书约需 70 学时,其中上机实验 20 学时。不讲第九章和第十章需要 50 学时,其中包括 14 学时上机实验。有些章节,如第三章,有大量的例题,不少例题属于阅读、理解、加深题,学时不充分时可以跳过不讲。

由于程序设计语言是个有机的整体,各部分内容之间相互联系、渗透,因此,读者在使用本书时,应该在学习后面章节的同时经常复习前面的内容,以便加深理解。

与本书配套的辅助教材《C 程序设计题解》也同时出版。

本书第二、三、四、九、十章由王柏盛编写,第五、八章由李万庆编写,第一、六、七章和全部附录由贺洪江编写。全书由王柏盛统稿、整理。

本书在编写、出版过程中得到了河北工程大学有关领导的关心和支持,在此深致谢意。

由于作者的水平有限、经验不足,编写时间仓促,书中难免存在许多缺点不足之处,恳请广大读者和同行批评指正。

王柏盛  
2003 年 12 月

# 目 录

第一章 C语言概述 .....	( 1 )
1.1 C语言的起源 .....	( 1 )
1.2 C语言的特点 .....	( 1 )
1.3 C语言的词法 .....	( 2 )
1.3.1 字符集 .....	( 2 )
1.3.2 关键字 .....	( 3 )
1.3.3 标识符 .....	( 3 )
1.4 C程序的组成和结构特点 .....	( 4 )
1.4.1 程序举例 .....	( 4 )
1.4.2 结构特点 .....	( 6 )
1.5 C程序的编辑、编译、连接和运行 .....	( 7 )
1.5.1 C源程序的编辑 .....	( 7 )
1.5.2 C源程序的编译和连接 .....	( 7 )
1.5.3 Turbo C的内存映射 .....	( 7 )
1.5.4 C源程序的调试过程 .....	( 8 )
1.6 标准输入/输出函数 .....	( 8 )
1.6.1 格式化输入/输出函数 .....	( 8 )
1.6.2 非格式化输入/输出函数 .....	( 14 )
习题一 .....	( 17 )
实验一 Turbo C源程序的编辑、编译、调试和运行 .....	( 20 )
第二章 常量、变量、运算符和表达式 .....	( 21 )
2.1 数据类型 .....	( 21 )
2.2 常量 .....	( 22 )
2.2.1 常量的数据类型 .....	( 22 )
2.2.2 常量的表示方法 .....	( 22 )
2.3 变量 .....	( 23 )
2.3.1 变量的类型 .....	( 23 )
2.3.2 变量的定义 .....	( 23 )
2.3.3 变量的作用域 .....	( 25 )
2.3.4 变量的存储类型 .....	( 28 )
2.3.5 变量的初始化 .....	( 32 )
2.4 数组 .....	( 33 )

---

2.4.1	数组的定义.....	(33)
2.4.2	数组的引用.....	(34)
2.4.3	数组的初始化.....	(35)
2.4.4	应用举例.....	(37)
2.5	指针 .....	(39)
2.6	运算符和表达式 .....	(39)
2.6.1	算术运算符和加 1、减 1 运算符 .....	(39)
2.6.2	关系运算符、逻辑运算符及其表达式 .....	(42)
2.6.3	按位运算符和位运算表达式.....	(43)
2.6.4	特殊运算符及其表达式.....	(48)
2.6.5	运算符优先顺序和结合性.....	(51)
2.7	表达式的计算过程和数据类型转换 .....	(52)
2.7.1	表达式的计算过程.....	(52)
2.7.2	表达式中的类型转换.....	(55)
2.7.3	程序举例.....	(57)
2.8	综合举例 .....	(60)
	习题二 .....	(66)
	实验二 基本输入/输出函数和运算符、表达式 .....	(72)
第三章	程序控制语句 .....	(73)
3.1	C 语句概述.....	(73)
3.1.1	C 程序结构.....	(73)
3.1.2	语句分类.....	(73)
3.2	结构化程序基本结构 .....	(75)
3.2.1	顺序结构.....	(75)
3.2.2	选择结构.....	(75)
3.2.3	循环结构.....	(76)
3.3	顺序结构程序设计语句 .....	(77)
3.4	分支结构程序设计语句 .....	(78)
3.4.1	if 语句 .....	(78)
3.4.2	switch 语句 .....	(85)
3.5	循环结构程序设计语句 .....	(92)
3.5.1	goto 语句以及用 goto 语句和 if 语句构成循环 .....	(92)
3.5.2	while 语句 .....	(94)
3.5.3	do while 语句 .....	(94)
3.5.4	for 语句 .....	(95)
3.5.5	循环的嵌套.....	(98)
3.5.6	几种循环的比较.....	(98)

3.5.7 程序举例.....	( 98 )
3.6 break 和 continue 语句 .....	( 104 )
3.6.1 break 语句 .....	( 104 )
3.6.2 continue 语句 .....	( 105 )
3.6.3 程序举例 .....	( 105 )
3.7 return 语句和 exit( )函数调用语句 .....	( 107 )
3.7.1 return 语句 .....	( 107 )
3.7.2 exit( )函数调用语句 .....	( 107 )
3.8 综合举例 .....	( 108 )
习题三 .....	( 127 )
实验三(1) 分支结构程序设计 .....	( 138 )
实验三(2) 循环结构程序设计 .....	( 138 )
第四章 函数 .....	( 139 )
4.1 函数的定义 .....	( 139 )
4.1.1 定义形式 .....	( 139 )
4.1.2 使用说明 .....	( 140 )
4.1.3 应用举例 .....	( 142 )
4.1.4 Turbo C 函数的扩展定义 .....	( 142 )
4.2 函数的调用 .....	( 144 )
4.2.1 调用形式 .....	( 144 )
4.2.2 调用过程 .....	( 145 )
4.2.3 调用条件 .....	( 146 )
4.2.4 嵌套调用 .....	( 147 )
4.3 函数间的数据传递 .....	( 150 )
4.3.1 传值方式传递数据 .....	( 151 )
4.3.2 传址方式传递数据 .....	( 152 )
4.3.3 利用全局变量传递数据 .....	( 153 )
4.3.4 处理结果在函数间的传递 .....	( 154 )
4.4 函数与数组 .....	( 154 )
4.5 递归函数 .....	( 156 )
4.6 综合举例 .....	( 160 )
习题四 .....	( 173 )
实验四 函数 .....	( 177 )
第五章 指针 .....	( 178 )
5.1 指针变量的定义和初始化 .....	( 178 )
5.1.1 指针的概念 .....	( 178 )
5.1.2 指针变量的定义 .....	( 180 )

---

5.1.3	指针变量的初始化 .....	( 181 )
5.1.4	近程指针和远程指针 .....	( 182 )
5.2	指针运算 .....	( 182 )
5.2.1	取地址运算 .....	( 182 )
5.2.2	赋值运算 .....	( 183 )
5.2.3	取内容运算 .....	( 183 )
5.2.4	指针的算术运算 .....	( 184 )
5.2.5	关系运算 .....	( 186 )
5.3	指针与数组 .....	( 187 )
5.3.1	指向数组元素的指针变量的定义和引用 .....	( 187 )
5.3.2	指向多维数组的指针变量 .....	( 190 )
5.3.3	字符串的指针变量 .....	( 192 )
5.4	指针和函数 .....	( 196 )
5.4.1	用指针作为函数的参数 .....	( 196 )
5.4.2	指向函数的指针变量 .....	( 201 )
5.4.3	指针型函数 .....	( 206 )
5.5	指针数组和多级指针 .....	( 209 )
5.5.1	指针数组 .....	( 209 )
5.5.2	指针的指针 .....	( 211 )
5.5.3	指针数组作主函数的形参 .....	( 212 )
5.6	程序举例 .....	( 214 )
	习题五 .....	( 219 )
	实验五 指针 .....	( 227 )
第六章	结构、联合、枚举和定义类型 .....	( 228 )
6.1	结构 .....	( 228 )
6.1.1	结构的说明 .....	( 228 )
6.1.2	结构变量的定义 .....	( 229 )
6.1.3	结构成员的引用 .....	( 232 )
6.1.4	结构变量的初始化 .....	( 233 )
6.1.5	指向结构的指针 .....	( 236 )
6.1.6	用指向结构的指针作为函数参数 .....	( 238 )
6.1.7	结构型函数和结构指针型函数 .....	( 242 )
6.1.8	动态数据结构 .....	( 246 )
6.1.9	位域结构 .....	( 249 )
6.2	联合 .....	( 252 )
6.2.1	联合说明和联合变量的定义 .....	( 252 )
6.2.2	联合变量的引用方式 .....	( 253 )

6.2.3 联合类型数据的特点 .....	(254)
6.2.4 应用举例 .....	(256)
6.3 枚举 .....	(258)
6.4 定义类型 .....	(261)
习题六 .....	(263)
实验六 结构、联合、枚举 .....	(266)
第七章 编译预处理命令 .....	(267)
7.1 宏定义 .....	(267)
7.1.1 不带参数的宏定义 .....	(267)
7.1.2 带参数的宏定义 .....	(270)
7.2 文件包含 .....	(275)
7.3 条件编译 .....	(276)
习题七 .....	(279)
实验七 编译预处理命令 .....	(281)
第八章 文件 .....	(282)
8.1 文件概述 .....	(282)
8.1.1 流和文件 .....	(282)
8.1.2 标准设备文件的换向和管道连接 .....	(284)
8.1.3 控制台输入/输出函数 .....	(287)
8.2 文件类型指针 .....	(288)
8.3 文件的打开与关闭 .....	(289)
8.3.1 文件的打开( <code>fopen()</code> 函数 ) .....	(289)
8.3.2 文件的关闭( <code>fclose()</code> 函数 ) .....	(291)
8.4 文件结束检测及出错检测 .....	(291)
8.4.1 <code>feof()</code> 函数 .....	(291)
8.4.2 <code>ferror()</code> 函数 .....	(292)
8.5 文件的读/写 .....	(292)
8.5.1 <code>fputc()</code> 函数和 <code>fgetc()</code> 函数( <code>putc()</code> 函数和 <code>getc()</code> 函数 ) .....	(292)
8.5.2 <code>fread()</code> 函数和 <code>fwrite()</code> 函数 .....	(295)
8.5.3 <code>fprint()</code> 函数和 <code>fscan()</code> 函数 .....	(297)
8.5.4 其他读/写函数 .....	(298)
8.6 文件的定位 .....	(300)
8.6.1 <code>rewind()</code> 函数 .....	(300)
8.6.2 <code>fseek()</code> 函数 .....	(300)
8.6.3 <code>fseek()</code> 函数 .....	(301)
8.7 非缓冲文件系统 .....	(302)
8.7.1 <code>open()</code> 、 <code>creat()</code> 和 <code>close()</code> 函数 .....	(302)

8.7.2 read( )和 write( )函数 .....	( 304 )
8.7.3 lseek( )函数和 tell( )函数 .....	( 305 )
8.8 小结 .....	( 305 )
习题八 .....	( 306 )
实验八 文件 .....	( 308 )
第九章 字符屏幕和图形函数 .....	( 309 )
9.1 PC 图形适配器及其工作模式 .....	( 309 )
9.2 字符屏幕函数 .....	( 310 )
9.2.1 窗口 .....	( 310 )
9.2.2 基本输入/输出函数 .....	( 310 )
9.2.3 屏幕操作函数 .....	( 311 )
9.2.4 字符属性控制函数 .....	( 315 )
9.2.5 字符屏显状态函数 .....	( 318 )
9.2.6 directvideo 变量 .....	( 320 )
9.2.7 演示程序 .....	( 320 )
9.3 Turbo C 的图形函数 .....	( 321 )
9.3.1 图形模式的初始化 .....	( 322 )
9.3.2 屏幕颜色的设置和清屏函数 .....	( 325 )
9.3.3 基本图形函数 .....	( 327 )
9.3.4 封闭图形的填充 .....	( 332 )
9.3.5 有关图形视口和图形操作函数 .....	( 337 )
9.3.6 图形模式下的文本输出 .....	( 341 )
9.3.7 独立图形运行程序的建立 .....	( 345 )
习题九 .....	( 346 )
实验九 字符屏幕和图形函数 .....	( 347 )
第十章 实用编程技术 .....	( 348 )
10.1 Turbo C 库函数介绍 .....	( 348 )
10.1.1 库文件的概念 .....	( 348 )
10.1.2 Turbo C 提供的 BIOS、DOS 系统调用函数 .....	( 350 )
10.1.3 日期和时间函数 .....	( 363 )
10.1.4 字符串函数、数字字符串与数值的转换函数 .....	( 367 )
10.1.5 动态内存分配函数、过程控制和数学运算函数 .....	( 370 )
10.2 Turbo C 的存储模式 .....	( 374 )
10.2.1 Turbo C 的存储模式 .....	( 374 )
10.2.2 编译程序的内存模式选择 .....	( 376 )
10.2.3 混合模式编程 .....	( 376 )
10.2.4 Turbo C 的段修饰符 .....	( 378 )

---

10.3 Turbo C 集成开发环境下程序的调试 .....	( 378 )
10.3.1 编译时的常见错误 .....	( 378 )
10.3.2 连接时的常见错误 .....	( 379 )
10.3.3 运行时的常见错误 .....	( 379 )
10.4 Turbo C 的命令行编译 .....	( 380 )
10.5 Turbo C 中汉字的使用 .....	( 382 )
10.5.1 汉字操作系统下汉字输入/输出的程序编制 .....	( 382 )
10.5.2 非汉字操作系统下汉字的使用 .....	( 385 )
10.6 Turbo C 和汇编程序的接口 .....	( 396 )
10.6.1 Turbo C 调用汇编子程序 .....	( 396 )
10.6.2 Turbo C 行间嵌入汇编 .....	( 398 )
10.7 Turbo C 2.0 集成开发环境的安装和使用 .....	( 401 )
10.7.1 Turbo C 2.0 软盘内容简介 .....	( 401 )
10.7.2 Turbo C 2.0 的安装和启动 .....	( 401 )
10.7.3 Turbo C 2.0 集成开发环境的使用 .....	( 402 )
10.7.4 Turbo C 的配置文件的配置 .....	( 410 )
附录 .....	( 412 )
附录一 常用字符与 ASCII 码对照表 .....	( 412 )
附录二 C 语言中的关键字 .....	( 413 )
附录三 运算符和优先级 .....	( 413 )
附录四 C 语言常用语法提要 .....	( 414 )
附录五 Turbo C 常用库函数表 .....	( 419 )
附录六 键盘扩展码表 .....	( 438 )
参考文献 .....	( 439 )

# 第一章 C 语言概述

## 1.1 C 语言的起源

C 语言是 1972 年由美国人丹尼斯·里奇( Dennis Ritchie )设计发明的 ,并首次在 UNIX 操作系统的 DECPDP - 11 计算机上使用。它是由早期的编程语言 BCPL ( Basic Combined Programming Language )发展演变而来的。在 1970 年 ,AT&T 贝尔实验室的肯·汤普森( Ken Thompson )根据 BCPL 语言设计出较先进的语言 ,即 B 语言 ,在此基础上促进了 70 年代 C 语言的问世。

随着微型计算机的日益普及 ,C 语言出现了多种版本。为了改变这些版本不一致的情况 ,1983 年美国国家标准学会为 C 语言制定了一套 ANSI 标准。Turbo C 完全是按照 ANSI 的 C 语言标准实施的。它是一种快速、高效的编译程序。Turbo C 不仅提供一个集成开发环境 ,而且还按传统方式提供了一个命令行编译程序版本 ,以满足不同用户的需要。

## 1.2 C 语言的特点

### 1. 结构化语言

结构化语言的一个显著特点是代码和数据的分隔化 ,即程序的各部分除了必要的信息交流外彼此互不影响 ,相互隔离。C 语言的主要结构成分是函数 ,函数是 C 语言的基本结构模块 ,所有的程序活动内容都包含在其中 ,函数在程序中被定义完成独立的任务 ,独立地编译成目标代码 ,这样可以实现程序的模块化。C 语言中 ,另一个实现程序结构化和分离化的方法是使用复合语句 ,复合语句是作为一个语句对待的 ,且具有逻辑联系的程序语句的组合 ,它是一个逻辑单元。严格地说 ,C 语言并不是一种真正的结构化语言 ,因为它不允许在子程序或函数中再定义子程序或函数 ,但由于它的结构类似于 ALGOL、Pascal 和 Modula - 2 ,通常还是把 C 语言称为结构化语言。因为 ,C 语言有现代程序设计语言的各种数据结构 ,它的数据类型有整型、实型、字符型、数组、结构、联合及指针和无值型 ,能用来实现各种复杂

的数据结构的运算,如队列、链表、树、图、堆栈等。另外,C语言具有结构化的控制语句,如if...else、while、do...while、for、switch等语句。C语言用函数作为程序模块,以实现程序的模块化,是结构化的编程语言,符合现代编程风格。

## 2. 简洁、紧凑、灵活

Turbo C共有43个关键字,9种控制语句,程序书写自由,主要以小写字母表示,压缩了一切不必要的成分。C语言语法限制不太严格,程序设计自由度大,例如对数组边界不做检查,整型、字符型、逻辑型数据可以通用。

## 3. 运算符丰富

C语言共有44种运算符,把括号、赋值、强制类型转换等都作为运算符处理,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。

## 4. 中级语言

C语言把高级语言的基本结构与低级语言的实用性结合起来。它可以像汇编语言一样对位、字节和地址进行操作,这三者是计算机最基本的工作单元,C语言可实现汇编语言的绝大部分功能。

## 5. 移植性好

用C语言编写的程序移植性好,生成的目标代码质量高,程序执行速度快。

## 6. 功能强大

C语言有丰富的库函数、强大的图形功能,有预处理能力,和其他语言如汇编语言、Pascal语言、数据库语言等的接口容易实现,C程序中还可以直接调用DOS命令。因此,C语言适合于编写各种系统软件、工具软件等大型软件。目前,在工业计算机控制系统开发中,越来越多的人都使用C语言来编写控制软件。

## 7. 编译语言

C语言是编译语言,用C语言写的源程序必须经过编译后才能运行。

# 1.3 C 语言的词法

## 1.3.1 字符集

C语言的字符集包括:大小写英文字母、数字、下划线“\_”及+、-、\*、/、=、<、>、?、!、%、&、^、#、'、<、>、|、(、)、{、}、[、]、~、\等。

### 1.3.2 关键字

关键字又称为保留字。C 语言编译系统对关键字赋有专门的含义。Turbo C 共有 43 个关键字,分别包括以下几种类型:

- (1) 描述存储属性的有 :auto ,extern ,static ,register ;
- (2) 描述数据类型的有 :char ,int ,float ,double ,void ,struct ,union ,enum ,long ,short ,signed ,unsigned ;
- (3) 描述语句的有 :goto ,if ,else ,switch ,case ,default ,break ,for ,do ,while ,continue ,return ;
- (4) 描述访问方式的有 :const( 常量修饰符) ,volatile( 易变量修饰符) ;
- (5) 描述编译状态的有 :sizeof ;
- (6) 描述数据类型定义的有 :typedef ;
- (7) Turbo C 扩展的关键字有 :asm ,\_cs ,\_ds ,\_es ,\_ss ,\_cdecl ,pascal ,far ,near ,huge ,interrupt。

### 1.3.3 标识符

C 语句中以标识符命名程序中的对象名,如函数、变量、符号常量、数组、结构、联合、指针、数据类型、存储属性、标号及宏等。

标识符是以字母、数字和下划线组成的,但第一个字符必须是字母或下划线。Turbo C 规定标识符的有效字符长度为 1~32 个字符。在 C 语言中字母大小写是有区别的, COUNT、Count 和 count 为 3 个不同的标识符。习惯上,符号常量、宏名等用大写字母,变量、函数名等用小写字母,系统变量由下划线开头。

标识符不能和 Turbo C 中的关键字相同,虽然 Turbo C 并不排斥标识符和库函数中的函数名相同,但程序中应该尽量避免用库函数名做标识符。另外,main 虽然不是 ANSI 标准的关键字,但各种 C 语言版本都把它作为主函数名,因此也不应该把它作为标识符。C 源程序名不属于 C 语句,属于操作系统,其命名要求符合操作系统文件名的规定,其扩展名通常为 .C。

标识符的选择应该做到“见名知义”、“常用取简”、“专用取繁”。例如 :count、name、year、month、student\_number、display、screen\_format 等,使之一目了然,以增加程序的可读性。

例:

正确的标识符名	不正确的标识符名
test_123	100
	数字打头

_label_100	integer !	包含 ! 字符
rectangle_area	chinese word	包含空格字符
circle_radius_r	for	是关键字
students	printf	虽然可用 ,但与库函数中的输出函数同名

## 1.4 C 程序的组成和结构特点

C 语言的 43 个关键字再加上语法规则 ,就构成了 C 编程语言。所有的 C 语言关键字都是小写字母。

所有的 C 语言程序都包含一个或多个函数。惟一不可缺少的是 `main( )` 函数 ,它是程序开始运行时第一个被调用的函数。一个好的 C 语言程序中 ,`main( )` 函数提纲性地列出程序所要做的事情 ,而这一提纲由一系列函数调用所组成。`main` 虽然不是 C 语言的组成部分 ,但还是应该把它作为关键字来对待 ,不要把它用作变量名 ,否则有可能使编译程序产生混乱。

### 1.4.1 程序举例

#### 【例 1.1】

```
main( )
{
    printf( "This is a C Program.\n" );
}
```

本程序的作用是输出以下一行信息 :

This is a C Program.

其中 `main` 表示“主函数” ,每一个 C 程序都必须有一个 `main` 函数。函数体由一对大括号 `{ }` 括起来。本例中主函数的函数体只包括一条输出语句 ,`printf` 是 C 语言中的输出函数 ,双引号中的字符串原样输出。“`\n`”是换行符 ,即在输出信息“`This is a C Program.`”后回车换行。语句的最后有一个分号 ,分号是语句的组成部分。

#### 【例 1.2】

```
main( )                /* 求两数之和 */
{
    int a , b , sum ;    /* 定义变量 */
    a = 123 , b = 456 ;
    sum = a + b ;
```

```
    printf( "sum is %d \n" ,sum );
}
```

本程序的作用是求两个整数  $a$  和  $b$  之和  $sum$ 。 `/* ... */` 表示注释部分, 注释只是为了方便阅读而加的, 对编译和运行不起作用。第二行是变量定义部分, 说明  $a$ 、 $b$  和  $sum$  是整型变量( `int` )。第三行是两条赋值语句, 使  $a$  和  $b$  的值分别是 123 和 456。第四行计算  $a + b$  的和并赋给  $sum$ 。第五行的 `%d` 表示按十进制整数格式输出  $sum$  的值。本例编译运行后输出的信息为

```
sum is 579
```

### 【例 1.3】

```
main( )                /* 主函数 */
{
    int a ,b ,c ;      /* 定义变量 */
    scanf( "%d ,%d" &a ,&b ); /* 输入 a 和 b 的值 */
    c = max( a ,b );  /* 调用函数 max ,将得到的值赋给 c */
    printf( "max = %d" ,c ); /* 输出 c 的值 */
}

int max( x ,y )       /* 定义 max( )函数 ,函数值为整型 ,x、y 为形式参数 */
int x ,y ;           /* 定义形参 x、y 为整型 */
{
    int z ;          /* 定义 max( )函数中的 z 为整型变量 */
    if( x > y ) z = x ;else z = y ; /* 求 x、y 的最大值并赋给 z */
    return( z );    /* 从子函数返回调用处 ,z 为返回的函数值 */
}
```

本程序包含两个函数: 主调函数 `main( )` 和被调函数 `max( )`。 `max( )` 的作用是将  $x$  和  $y$  中的较大值赋给  $z$ 。 `return` 语句将  $z$  的值返回给主调函数 `main( )`。返回值是通过函数名 `max( )` 带回到 `main( )` 的调用处。 `main( )` 函数中的 `scanf` 是“输入函数”的名字, 其作用是通过键盘输入  $a$  和  $b$  的值。 `&a` 和 `&b` 中的 `&` 的含义是“取变量的地址”; `%d ,%d` 的含义按十进制整数形式输入。 `main( )` 函数中的第三行为调用 `max( )` 函数, 在调用时将实际参数  $a$  和  $b$  的值分别传送给 `max( )` 函数中的形式参数  $x$  和  $y$ 。 执行函数 `max` 后, 得到一个返回值赋给主函数中的变量  $c$ 。 本程序编译运行结果为:

```
输入 8 5 < Enter >      输入 8 和 5 给 a 和 b , < Enter > 为回车键
输出 :max = 8           输出 c 的值
```

本例中用到了函数调用、实参和形参等概念, 将在函数一章中再详细介绍。

## 1.4.2 结构特点

通过上述几个简单的例子,可以看出以下几点:

(1) C 程序是由函数构成的。一个 C 程序中必须有一个名为 main 的主函数和若干个子函数(可以没有)。函数是 C 程序的基本单位。被调函数可以是系统提供的库函数,如 printf 和 scanf 函数,也可以是用户自己编写的函数,如例 1.3 中的 max 函数。C 函数相当于其他语言中的子程序。C 语言用函数来实现特定功能,可以说 C 语言是函数式的语言,程序的全部工作都是由函数来完成的。C 语言函数库十分丰富, Turbo C 提供三百多个库函数。这种特点使得 C 语言容易实现程序模块化。

当一个程序较大、包括很多函数时,可以分成若干个文件,每个文件包括几个函数。

(2) 一个函数由两部分组成:① 函数说明部分,包括函数名、函数类型、函数属性、函数参数和参数类型,函数名后必须跟一对圆括号,可以没有函数参数。② 函数体,即大括号内的部分,如果函数内有多对大括号,则最外层的一对大括号为函数体的范围。函数体一般包括变量定义和执行部分,也可以没有变量定义部分,甚至连执行部分也没有,这是一个空函数,它什么也不干。

(3) 一个 C 程序总是从 main 函数开始执行,与 main 函数在程序中的位置无关,即 main 函数可以在其他函数的前面或后面。

(4) C 程序书写格式自由,一行可以写一条语句,也可以写多条语句,一条语句也可以分成几行写,没有行号,也不像 FORTRAN 和 COBOL 那样严格规定书写格式。C 语言规定可以在任何一个分隔符和空格处换行。

(5) 每条语句以分号结束,分号是语句的组成部分,即使是最后一条语句分号也不可缺少。

(6) C 语言没有输入/输出语句,输入/输出是由库函数完成的,C 对输入/输出实行“函数化”。

(7) 可以用 /\* ... \*/ 对程序中的任何部分做注释,以增加程序的可读性,注释对编译和运行不起作用。/\* 和 \*/ 必须成对出现,且/与 \*、\* 与/之间不允许有其他字符(如空格)出现。注释不能插在关键字或标识符的中间,C 语言规定有分隔符的地方都可以插入注释,或者能换行的地方都可以插入注释。通常注释可以出现在程序段前面,以对该程序段加以说明,或在语句行的后面出现,以对该语句加以说明。

【注意】语句  $a = b/*p$  的含义是变量 b 除以 \*p 的商赋给变量 a,\*p 是指针变量 p 指向的变量,这种写法会引起编译错误,因为 C 编译系统把 /\* 看成是注释的起点,找到 \*/ 才认为注释结束。修改的办法是将语句  $a = b/*p$  改为  $a = b(*p)$ 。