



全国高等专科教育计算机类规划教材

C 语言程序设计

陈耀东 主编

 机械工业出版社
CHINA MACHINE PRESS



全国高等专科学校教育计算机类规划教材

C 语言程序设计

主 编 陈耀东
副主编 黄 勇 吴学毅
参 编 罗海勇 刘金花
 吴 媛 刘志杰
 杨春蓉



机械工业出版社

本书分为 11 章, 主要内容包括: C 语言概述、数据类型与表达式、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、编译预处理、指针、结构类型与其他构造类型、文件。

通过对本书的学习, 读者可以很快掌握 C 语言基础知识及其程序设计方法。本书力求通俗易懂, 可作为大、中专学生的高级语言程序设计课程教材, 也可作为 C 语言程序设计的自学教材及参考书。

本书配有电子教案供教师使用, 可发电子邮件至 wangyx@mail.machininfo.gov.cn 邮箱索取。

图书在版编目(CIP)数据

C 语言程序设计/陈耀东主编. —北京: 机械工业出版社, 2007. 1

全国高等专科教育计算机类规划教材

ISBN 7-111-20053-5

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

中国版本图书馆 CIP 数据核字(2006)第 122927 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 孔熹峻 责任编辑: 李学峰 版式设计: 张世琴
责任校对: 陈立辉 封面设计: 姚毅 责任印制: 李妍

北京铭成印刷有限公司印刷

2007 年 1 月第 1 版第 1 次印刷

184mm × 260mm · 14.75 印张 · 362 千字

0001—4000 册

定价: 22.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

本社购书热线电话(010)68326294

编辑热线电话(010)68354423

封面无防伪标均为盗版

前 言

C语言是一种应用十分广泛的程序设计语言，它功能丰富、表达能力强、使用方便灵活、程序执行效率高、可移植性好，既具有高级语言的优点，又具有低级语言的许多特点。C语言是学生们学习计算机课程的必修语言，而且也是一种通用的程序设计语言。现在流行的 Visual C++ 和 C++ Builder 等面向对象程序设计，都是以 C 语言为基础的语言。

主要内容：本书共分 11 章。第 1 章介绍了 C 语言概述和 Turbo C 使用初步，第 2 章介绍了 C 语言的数据类型与表达式，第 3 章介绍了顺序结构程序设计，第 4 章介绍了选择结构程序设计，第 5 章介绍了循环结构程序设计，第 6 章介绍了数组与字符数据处理，第 7 章介绍了函数与程序结构，第 8 章介绍了编译预处理的内容，第 9 章介绍了指针的基本内容，第 10 章介绍了结构类型与其他构造类型，第 11 章介绍了数据文件概念与操作。

适用对象：本书按照高等专科学校高级语言程序设计教学大纲的要求编写，在内容的组织上根据作者在 C 语言教学过程中遇到的问题和教学经验进行了有效的改进，突出了 C 语言的特点，对教材的内容作了精心的安排以适应高等专科学校学生的需求。

本书先介绍 C 语言的基本概念、语法以及 C 语言程序设计方法，以程序设计为基本线索，同时对 C 语言做深入介绍，强调如何认识程序、写程序和用 C 语言写程序。通过实例讨论问题的分析和分解，设计求解过程，找出主要步骤，确定函数抽象，找出循环，选择语言结构，最后写出程序的过程。并通过实例进行说明，将语法规则融入例题中加以讲解，一方面加强语法规则的理解，另一方面在学习语法的同时掌握 C 语言的应用，不仅强调 C 语言的基础知识，而且重视编程能力和上机调试能力的培养，充分调动学生的学习积极性。

本书可作为高等专科学校和高职相关专业的高级语言程序设计课程教材，也可作为 C 语言自学参考书。

编写分工：本书由陈耀东任主编，负责组稿、统稿；黄勇、吴学毅任副主编。其中第 1 章由罗海勇编写，第 2、6、9 章和附录由陈耀东编写，第 3 章由杨春蓉编写，第 4 章由吴媛编写，第 5、8 章由吴学毅编写，第 7 章由刘志杰编写，第 10 章由刘金花编写，第 11 章由黄勇编写，张志平、王春平、钟福连、邹国平参加了本书的部分习题编写。

由于时间仓促，加上水平有限，书中难免存在不足之处，欢迎读者批评指正。

编 者

目 录

前言

第 1 章 C 语言概述 1

- 1.1 C 语言的发展及其特点 1
 - 1.1.1 C 语言的发展 1
 - 1.1.2 C 语言的特点 1
- 1.2 C 语言的基本符号 2
- 1.3 简单的 C 语言程序 3
 - 1.3.1 几个简单的 C 语言程序 3
 - 1.3.2 C 语言程序的结构特点 5
- 1.4 C 语言程序的开发步骤 6
 - 1.4.1 C 语言程序开发的一般过程 6
 - 1.4.2 Turbo C 的初步使用 6
- 1.5 本章小结 8
- 习题 8

第 2 章 数据类型与表达式 9

- 2.1 数据与数据类型 9
- 2.2 C 语言中的基本数据类型 10
 - 2.2.1 常量与变量 10
 - 2.2.2 整型数据 11
 - 2.2.3 实型数据 14
 - 2.2.4 字符型数据 15
 - 2.2.5 变量初始化 18
- 2.3 算术运算与赋值运算 18
 - 2.3.1 C 语言中的运算规则 18
 - 2.3.2 算术运算符与算术表达式 20
 - 2.3.3 自增与自减运算 21
 - 2.3.4 赋值运算符和赋值表达式 22
- 2.4 关系运算与逻辑运算 24
 - 2.4.1 关系运算符与关系表达式 24
 - 2.4.2 逻辑运算符与逻辑表达式 25
 - 2.4.3 条件运算符与条件表达式 26
- 2.5 位运算 27

- 2.6 其他运算 30
 - 2.6.1 逗号运算符与逗号表达式 30
 - 2.6.2 sizeof 运算符 30
- 2.7 混合运算及数据类型转换 30
 - 2.7.1 混合运算 30
 - 2.7.2 数据类型转换 31
- 2.8 本章小结 32
- 习题 33

第 3 章 顺序结构程序设计 36

- 3.1 C 语句概述 36
- 3.2 结构化程序设计 37
- 3.3 基本输入输出操作的实现 40
 - 3.3.1 数据输入输出的概念及在 C 语言中的实现 40
 - 3.3.2 字符数据的输入输出 40
 - 3.3.3 格式的输入输出 41
- 3.4 程序举例 47
- 3.5 本章小结 49
- 习题 49

第 4 章 选择结构程序设计 52

- 4.1 if 语句 52
 - 4.1.1 简单 if 语句 52
 - 4.1.2 if() else 语句 53
 - 4.1.3 嵌套 if 语句 55
- 4.2 switch 语句 57
- 4.3 程序举例 59
- 4.4 本章小结 63
- 习题 63

第 5 章 循环结构程序设计 64

- 5.1 while 语句 65
- 5.2 do-while 语句 67

5.3 for 语句	68	7.3.1 函数的嵌套调用	116
5.4 break 语句和 continue 语句	71	7.3.2 函数的递归调用	117
5.5 循环的嵌套	73	7.4 数组作为函数参数	120
5.6 程序举例	77	7.4.1 数组元素作为函数的参数	120
5.7 本章小结	82	7.4.2 数组名作为函数的参数	121
习题	83	7.5 内部变量与外部变量	126
第 6 章 数组	85	7.5.1 内部变量	126
6.1 一维数组	85	7.5.2 外部变量	127
6.1.1 一维数组的定义	85	7.6 变量的存储方式	130
6.1.2 一维数组的引用	86	7.6.1 内部变量的存储方式	130
6.1.3 一维数组的初始化	86	7.6.2 外部变量的存储方式	133
6.1.4 一维数组程序举例	87	7.6.3 变量存储方式的总结	135
6.2 二维数组	95	7.7 内部函数和外部函数	135
6.2.1 二维数组的定义	95	7.8 工程文件	136
6.2.2 二维数组的引用	96	7.9 本章小结	137
6.2.3 二维数组的初始化	96	习题	138
6.2.4 二维数组程序举例	97	第 8 章 编译预处理	140
6.3 字符数组	98	8.1 宏定义	140
6.3.1 字符数组的定义	98	8.1.1 无参数的宏定义	140
6.3.2 字符数组的初始化	98	8.1.2 有参数的宏定义	142
6.3.3 字符数组的引用	99	8.2 文件包含	145
6.3.4 字符串的输入输出	99	8.3 条件编译	147
6.3.5 字符串处理函数	101	8.4 本章小结	148
6.3.6 字符数组程序举例	103	习题	149
6.4 本章小结	105	第 9 章 指针	151
习题	105	9.1 指针的基本概念	151
第 7 章 函数	108	9.1.1 指针与指针变量	151
7.1 C 语言程序的模块化	108	9.1.2 指针变量的说明	153
7.1.1 概述	108	9.1.3 指针的引用和运算	153
7.1.2 函数的种类	108	9.1.4 指针变量的初始化	154
7.2 函数的定义和调用	109	9.1.5 指针变量使用举例	154
7.2.1 函数的定义	109	9.2 指针与数组	156
7.2.2 函数的返回值及类型	110	9.2.1 通过指针引用一维数组中 的元素	156
7.2.3 函数的调用	112	9.2.2 通过指针引用二维数组中 的元素	163
7.2.4 对被调函数的声明	113	9.3 指针与函数	164
7.2.5 函数的形参与实参	114		
7.3 函数的嵌套调用与递归调用	116		

9.3.1 指针变量作函数参数	164	10.6 用 typedef 定义类型	203
9.3.2 再论数组名作函数参数	168	10.7 本章小结	204
9.3.3 返回指针值的函数	175	习题	204
9.3.4 函数的指针和指向函数的 指针变量	175	第 11 章 文件	207
9.4 指针与字符串	177	11.1 C 文件概述	207
9.4.1 字符数组与字符指针	177	11.2 文件指针	208
9.4.2 常见的字符串操作	177	11.3 文件的打开和关闭	209
9.5 指针数组与多级指针	180	11.3.1 文件的打开	209
9.5.1 指针数组	180	11.3.2 文件的关闭	210
9.5.2 主函数的参数	181	11.4 文件的读/写	210
9.5.3 多级指针	182	11.4.1 读/写文件中的一个字符	210
9.6 本章小结	183	11.4.2 读/写文件中的一个字符串	212
习题	183	11.4.3 读/写文件中的一个数据块	214
第 10 章 结构类型与其他构造类型	186	11.4.4 对文件进行格式化读/写	217
10.1 结构体类型	186	11.5 文件的定位	218
10.1.1 结构体类型的定义	187	11.5.1 位置指针复位函数	218
10.1.2 结构体类型变量的定义	188	11.5.2 改变文件位置指针函数	218
10.1.3 结构体变量的引用	190	11.5.3 返回位置指针当前值函数	219
10.1.4 结构体变量的初始化	190	11.6 本章小结	219
10.2 结构体数组	191	习题	220
10.2.1 结构体类型数组的定义	192	附录	222
10.2.2 结构体数组的初始化	192	附录 A 常用字符和 ASCII 码对照表	222
10.3 结构指针	194	附录 B 运算符和结合性	223
10.3.1 指向结构体变量的指针	194	附录 C C 语言中的关键字	224
10.3.2 指向结构体数组的指针	196	附录 D Turbo C 常用库函数	224
10.4 共用体和枚举型	196	参考文献	228
10.4.1 共用体	196		
10.4.2 枚举型	200		
10.5 位段简介	202		

第 1 章 C 语言概述

学习目标

- 1) 了解 C 语言的起源和特点。
- 2) 了解 C 语言的基本符号。
- 3) 掌握 C 语言的程序结构。
- 4) 熟悉 C 语言程序的开发步骤。

1.1 C 语言的发展及其特点

1.1.1 C 语言的发展

C 语言的发展如此迅速，而且成为最受欢迎的语言之一，主要是因为它具有强大的功能。许多著名的系统软件，如 DBASE III PLUS、DBASE IV 都是用 C 语言编写的。它既可以作为系统软件的开发工具，又可以用于开发应用软件。

C 语言是 1972 年由美国的 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。它由早期的编程语言 BCPL (Basic Combined Programming Language) 发展演变而来。1970 年，AT&T 贝尔实验室的 Ken Thompson 根据 BCPL 语言设计出较先进的并取名为 B 的语言，1972 年贝尔实验室的布朗·W. 肯尼汉和丹尼斯·M. 利奇对 B 语言进行了完善和扩充，并取了“BCPL”的第二个字母“C”作为新语言的名称，这样 C 语言就问世了。

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，使得这些 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，称为 ANSI C，成为现行的 C 语言标准。

目前在微型计算机上使用的 C 语言版本有许多种，如 Microsoft C、Turbo C、Quick C、BORLAND C，本书以 Turbo C 为例讲解。

1.1.2 C 语言的特点

1. C 语言是一种中级语言

C 语言被认为是一种中级语言，这是因为它既包括高级语言的一些成分，又有汇编语言 (低级语言) 的一些功能。C 语言允许操纵计算机的基本要素，如位、字节、地址等，而且，C 语言代码的移植性非常好。也就是说，在一种类型的计算机上编写的软件可以很方便地在另一种类型的计算机上使用。C 语言虽然有五种基本的内置数据类型，但是与高级语言相比，C 语言并不是强大的类型语言。C 语言允许数据类型转换，它允许直接操纵位、字节以及指针。因此，C 语言可以用于系统级的程序设计。

2. C 语言是结构化的语言

术语“块结构语言”不适用于 C 语言。块结构语言允许在一个过程或函数内声明其他过程和函数，而 C 语言不允许在函数内创建函数，因此不是块结构语言。然而，由于 C 语言在许多方面与其他结构化语言 (Algol、Pascal 以及其他类似的语言) 相似，所以被认为是一种结构化语言。C 语言允许分割代码与数据，这种特性与其他结构化语言相同，这种特性能够将执行特定任务的信息和指令收集起来，从而禁止程序的其他部分访问这部分内容。这种特性可以通过函数或代码块来实现。函数用于定义和划分程序所需要的任务，这样程序可以作为一个单元使用。代码块是逻辑上连接在一起的一组程序语句，可以当作一个单元使用。将一系列语句放在一对花括号中，就形成了一个代码块，例如：

```
do {  
    i = i + 1;  
    :  
} while(i < 40);
```

1.2 C 语言的基本符号

在 C 语言中使用的词汇分为六类：标识符、关键字、运算符、分隔符、常量和注释符。

(1) 运算符 C 语言中含有相当丰富的运算符。运算符与变量、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。

(2) 分隔符 在 C 语言中采用的分隔符有逗号和空格两种。逗号主要用在类型说明和函数参数表中，分隔各个变量。空格多用于语句各单词之间，作间隔符。在关键字、标识符之间必须要有一个以上的空格符作间隔，否则将会出现语法错误。例如把 `int a` 写成 `inta`，C 编译器会把 `inta` 当成一个标识符处理，其结果必然出错。

(3) 常量 C 语言中使用的常量可分为数字常量、字符常量、字符串常量、符号常量、转义字符等多种，在第 2 章中将专门给予介绍。

(4) 注释符 C 语言的注释符是以“/*”开头并以“*/”结尾的串。在“/*”和“*/”之间的即为注释。程序编译时，不对注释作任何处理。注释可出现在程序中的任何位置。注释用来向用户提示或解释程序的意义。在调试程序中对暂不使用的语句也可用注释符括起来，使编译跳过不作处理，待调试结束后再去掉注释符。在 VC 编辑器中，如果注释仅包含一行，只需在注释语句之前使用 // 来表明这是一行注释。例如：

```
int a = 0; //This is a sample program
```

(5) 标识符 就像每个人有不同的名字一样，在 C 语言程序中，为了区别各个变量、函数、标签和其他用户定义的对象，都必须为它们取不同的名字，这些名字称为标识符。C 语言规定，标识符的第一个字符必须是字母或下划线(_)，后面的字符可以是字母、数字或者下划线。

以下标识符是合法的：

```
a、s_sum、marks3、num_1、BOOK1
```

以下标识符是非法的：

```
1a(以数字开头)、s*t(出现非法字符*)、book-1(出现非法字符-)
```

标识符的长度可以是任何合适的值，但是各种编译器能够识别的标识符字符数目有所不同。例如，MSC(Microsoft C)规定标识符前八位有效，当两个标识符前八位相同时，则被认为是同一个标识符。Turbo C 则可以识别 32 个字符。

在 C 语言中使用标识符时必须注意以下几点：

- 1) 标识符必须以字母或下划线开头。
- 2) 第一个字符后面可以跟一系列的字母或者数字，也可以包括一个特殊字符_。
- 3) C 语言中的标识符区分大小写，也就是说 sum、Sum 和 SUM 都是各不相同的标识符。
- 4) 标识符虽然可由程序员随意定义，但它应该描述出标识符所包含的数据的性质。例如，如果要计算两个数之和，那么用来存储这个结果的变量可以命名为 sum。把它命名为 abc 就不是很好。因此，命名应体现出相应的意义，以便于阅读理解，做到“见名知义”。

(6) 关键字 所有的语言都保留了一些单词供内部使用，这些单词在一个特定语言的上下文中含有特殊的意义，它们被称为“关键字”。在 C 语言中所有关键字必须小写。在 C 语言中命名标识符时，要确保没有使用任何关键字来作为标识符。

因此，在关键字、标识符之间必须有一个以上的空格符作间隔，否则将会出现语法错误。

C 语言的关键字分为以下几类：

- 1) 类型说明符。用于定义和说明变量、函数或其他数据结构的类型，如前面例题中用到的 int、double 等。
- 2) 语句定义符。用于表示一个语句的功能，如 if else 就是条件语句的语句定义符。
- 3) 预处理命令字。用于表示一个预处理命令，如 include。

1.3 简单的 C 语言程序

1.3.1 几个简单的 C 语言程序

下面先介绍几个简单的 C 语言程序，然后从中分析 C 语言程序的特性。

例 1-1 一个简单的 C 语言程序，用于输出一行信息。

```
main()                /* 定义主函数 */
{
    printf("This is a C program. \n");
}
```

本程序的作用是输出以下信息：

This is a C program.

其中 main 表示“主函数”，每一个 C 语言程序都必须有一个主函数。函数体由花括弧括起来。例 1-1 中主函数内只有一个输出语句，printf 是 C 语言中的输出函数(详见第 3 章)。双引号内的字符串按原样输出，“\n”是换行符，即在输出“This is a C program.”后回车换行，程序中的分号(;)是 C 语句的结束标志，C 语言规定每条语句都必须以分号结束。程序中“/* 定义主函数 */”是注释部分。

例 1-2 求两个数之和。

```
main()  
{ int a,b,sum;          /* 这是定义变量 */  
  a = 123;b = 456;      /* 以下 3 行为 C 语句 */  
  sum = a + b;  
  printf("sum = %d\n",sum);  
}
```

本程序的作用是求两个整数 a 和 b 之和 sum 。第二行是声明部分，定义变量 a 和 b ，指定 a 和 b 为整型 (int) 变量。第三行是两个赋值语句，使 a 和 b 的值分别为 123 和 456。第四行 sum 的值为 $a + b$ ，第五行中 “%d” 是输入输出的“格式字符串”，用来指定输入输出时的数据类型和格式 (详见第 3 章)，“%d” 表示“以十进制整数形式输出”。在执行输出时，此位置上代以一个十进制整数值。 $printf$ 函数中括弧内最右端 sum 是要输出的变量，现在它的值为 579 (即 $123 + 456$ 之值)，因此输出一行信息为：

```
sum = 579
```

例 1-3 求两个整数中的最大值，两个整数由键盘输入。

```
main()          /* 主函数完成两个整数的输入,这两个数中最大值输出 */  
{ int a,b,c;    /* 声明部分,定义变量 */  
  scanf("%d,%d",&a,&b); /* 输入变量 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);  
}  
  
int max(int x,int y) /* max 函数是求 x 和 y 的最大值 */  
{ int z;  
  if(x > y) z = x;  
  else z = y;  
  return(z);  
}
```

本程序包括两个函数：主函数和被调用的 max 函数。 max 函数的作用是将 x 和 y 中较大的值赋给变量 z 。 $return$ 语句将 z 的值返回给主函数。返回值是通过函数名 max 带回到主函数的调用处。主函数中的“ $scanf$ ”是“输入函数”的名字 ($scanf$ 和 $printf$ 都是 C 系统提供的标准输入输出函数)。程序中 $scanf$ 函数的作用是输入 a 和 b 的值。 $\&a$ 和 $\&b$ 中的“ $\&$ ”的含义是“取地址”，此 $scanf$ 函数的作用是将两个数值分别输入到变量 a 和 b 的地址所标志的单元中，也就是输入给变量 a 和 b 。 $\&a$ 和 $\&b$ 前面的“%d,%d”的含义与前面相同，只是现在用于“输入”，它指定输入的两个数值按十进制整数形式输入。关于 $scanf$ 函数详见第 3 章。

主函数中第四行为调用 max 函数，在调用时将实际参数 a 和 b 的值分别传送给 max 函数的形式参数 x 和 y 。经过执行 max 函数得到一个返回值 (即 max 函数中变量 z 的值)，把这个值赋给变量 c ，然后输出 c 的值， $printf$ 函数中的“ $max = \%d$ ”，在输出时，“%d”将由 c 的值取代，“ $max =$ ”原样输出。程序运行情况如下：

```
8,5↵          (输入 8 和 5 给 a 和 b,然后回车)
max = 8        (输出 c 的值)
```

本例用到了函数调用、实际参数和形式参数等概念，我们只做了很简单的解释。读者如对此不大理解，可以先不予以深究，在学到以后有关章节时，问题自然迎刃而解。介绍此例子，无非是使读者对 C 语言程序的组成和形式有一个初步的了解。

1.3.2 C 语言程序的结构特点

通过以上几个例子，可以看到：

(1) C 语言程序是由函数构成的 一个 C 语言源程序有且只有一个主函数，也可以包含一个主函数和若干个其他函数，因此，函数是 C 语言程序的基本单位。被调用的函数可以是系统提供的库函数(如 scanf 和 printf 函数)，也可以是用户根据需要自己编写设计的函数(如例 1-3 中的 max 函数)。C 语言中的函数相当于其他语言中的子程序，用函数来实现特定的功能。程序中的全部工作都是由各个函数分别完成的，编写 C 语言程序就是编写一个个函数。C 语言的函数库十分丰富，ANSI C 建议的标准库函数中包括 100 多个函数，Turbo C 和 MS C 4.0 提供 300 多个库函数。C 语言的这个特点使得容易实现程序的模块化。

(2) 一个函数由两部分组成

1) 函数的首部。即函数的第一行，包括函数类型、函数名、函数参数类型、函数参数(形参)名。

例如，例 1-3 中的 max 函数的首部为

```
int      max      (int      x,      int      y)
  ↓      ↓      ↓      ↓      ↓      ↓
函数类型 函数名  函数参数类型 函数参数名 函数参数类型 函数参数名
```

一个函数名后面必须跟一对圆括弧，函数参数则可以没有，如 main() 函数。

2) 函数体。即函数首部下面的花括号({……})内的部分，如果一个函数内有多个花括号，则最外层的{}为函数体的范围。

函数体一般包括：

① 声明部分。作用是定义所用到的变量，如例 1-3 中的主函数中的“int a,b,c;”。在第 7 章中还将看到，在声明部分中要对所调用的函数进行声明。

② 执行部分。由若干个语句组成，用以对数据进行操作，完成一定的算法处理。

当然，在某些情况下也可以没有声明部分(如例 1-1)，甚至可以既无声明部分，也无执行部分。例如：

```
dump()
{ }
```

它是一个空函数，什么也不干，但它是合法的。

3) 一个 C 语言程序总是从主函数开始执行，而不论主函数在整个程序中的位置如何(主函数可以放在程序最前头，也可以放在程序最后，或在一些函数之前，在另一些函数之后)。

4) C 语言程序书写格式自由，一行内可以写几个语句，一个语句可以分写在多行上。

5) 每个语句和数据定义的最后必须有一个分号，分号是 C 语言程序的必要组成部分

(如 $c = a + b;$)。分号不可少,即使是程序中最后一个语句也应包含分号(这和 PASCAL 是语言不同的)。

6) C 语言本身没有输入输出语句。输入和输出操作是由库函数 `scanf` 和 `printf` 等函数来完成的, C 语言对输入输出实行“函数化”。由于输入输出操作牵涉到具体的计算机设备,把输入输出操作放在函数中处理,就可以使 C 语言本身规模较小,编译程序简单,很容易在各种机器上实现,程序具有可移植性。当然,不同的 C 语言系统需要对函数库中的函数作不同的处理。不同的 C 语言系统除了提供函数库中的标准函数外,还按照硬件的情况提供一些专门的函数。因此不同的系统所提供的函数个数和功能是有所不同的。

7) 可以用“`/* …… */`”或“`//`”对 C 语言程序中的任何部分作注释。一个好的、有使用价值的源程序都应加上必要的注释,以增加程序的可读性。

1.4 C 语言程序的开发步骤

1.4.1 C 语言程序开发的一般过程

我们已经看到了一些用 C 语言编写的程序,为了使计算机能按照人们的意志进行工作,必须根据问题的要求,编写出相应的程序。所谓程序,就是一组计算机能识别和执行的指令,每一条指令使计算机执行特定的操作。程序可以用高级语言(如 QBASIC、FORTRAN、PASCAL、C 等)编写,用高级语言编写的程序称为“源程序”(Source Program)。从根本上说,计算机只能识别由 0 和 1 组成的二进制的指令,而不能识别和执行用高级语言写的指令。为了使计算机能执行高级语言源程序,必须先用一种称为“编译程序”的软件,把源程序翻译成二进制形式的“目标程序”,然后将该目标程序与系统的函数库和其他目标程序连接起来,形成可执行的目标程序。

在编好一个 C 语言源程序后,如何上机运行呢?在纸上写好一个程序后,要经过以下步骤:上机输入与编辑源程序→对源程序进行编译→与库函数连接→运行可执行的目标程序。例如,编辑后得到一个源程序文件 `f.c`,然后在进行编译时再将源程序文件 `f.c` 输入,经过编译得到目标程序文件 `f.obj`,现将目标程序文件 `f.obj` 输入内存,与系统提供的库函数等连接,得到可执行的目标程序文件 `f.exe`,最后把 `f.exe` 调入内存并使之运行。在了解 C 语言的初步知识后,读者最好上机运行一个 C 语言程序,以建立对 C 语言程序的初步认识。

1.4.2 Turbo C 的初步使用

1. Turbo C 概述

Turbo C 是美国 Borland 公司的产品, Borland 公司是一家专门从事软件开发、研制的公司。该公司相继推出了一套 Turbo 系列软件,如 Turbo BASIC, Turbo Pascal, Turbo Prolog, 这些软件很受用户欢迎。该公司在 1987 年首次推出 Turbo C 1.0 产品,使用了全新的集成开发环境,即使用一系列下拉式菜单将文本编辑、程序编译、连接以及程序运行一体化,大大方便了程序的开发。1988 年, Borland 公司又推出 Turbo C 1.5,增加了图形库和文本窗口函数库等,而 Turbo C 2.0 则是该公司 1989 年推出的,它在原来集成开发环境的基础上增加了查错功能,并可以在 Tiny 模式下直接生成 `.COM`(数据、代码、堆栈处在同一 64KB 内存中)文

件, 还可对数学协处理器(支持 8087/80287/80387 等)进行仿真。

Borland 公司后来又推出了面向对象的程序软件包 Turbo C + &127; + &127;, &127; &127; 继承发展了 Turbo C 2.0 的集成开发环境, 并包含了面向对象的基本思想和设计方法。

1991 年为了适用 Microsoft 公司的 Windows 3.0, Borland 公司又将 Turbo C ++ 作了更新, 即 Turbo C 的新一代产品 Borland C ++ 也已经问世了。

2. Turbo C 2.0 的基本配置

Turbo C 2.0 可运行于 IBM-PC 系列微型计算机, 包括 XT、AT 及 IBM 兼容机。此时要求 DOS 2.0 或更高版本支持, 并至少需要 448KB 的 RAM, 可在任何彩、单色 80 列监视器上运行, 它支持数学协处理器芯片, 也可进行浮点仿真, 这将加快程序的执行。

3. Turbo C 的安装与启动

Turbo C 2.0 的安装非常简单, 只要将光盘插入 A 驱动器中, 在 DOS 的 A > 下键入:

```
A > INSTALL
```

即可。只要在安装过程中按盘号的提示, 顺序插入各个软盘, 就可以顺利地进行安装, 安装完毕将在 C 盘根目录下建立一个 TC 子目录, TC 下还建立了两个子目录 LIB 和 INCLUDE, LIB 子目录中存放库文件, INCLUDE 子目录中存放所有头文件。

运行 Turbo C 2.0 时, 只要在 TC 子目录下键入 TC 并回车即可进入 Turbo C 2.0 集成开发环境。

说明: Turbo C 2.0 可以在互联网上下载, 解压后直接打开 TC. exe 文件即可运行 Turbo C 2.0, 如图 1-1 所示。

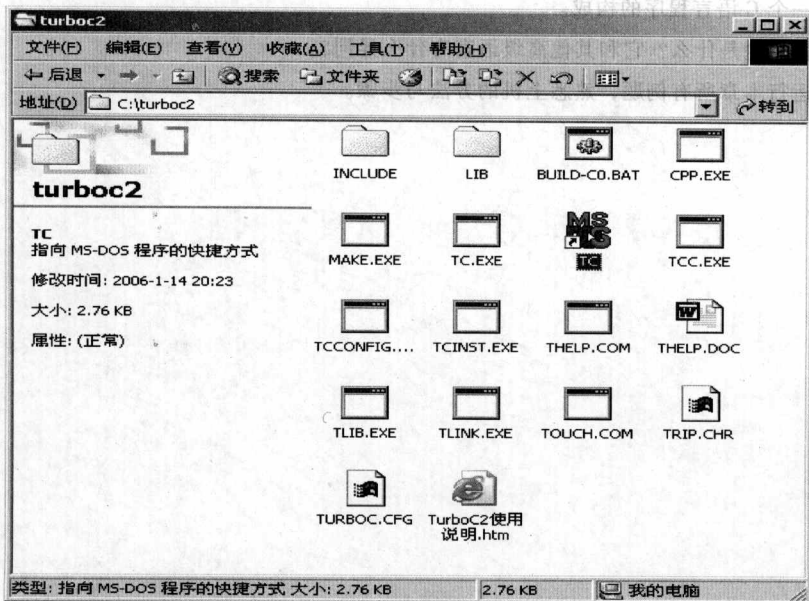


图 1-1 Turbo C 2.0 文件夹

有关 Turbo C 2.0 的使用将在配套的实验指导教材相应的章节中做详细的介绍。

1.5 本章小结

本章主要介绍了 C 语言的发展及其特点、C 语言的基本符号、C 语言程序的基本结构、C 语言程序的上机过程以及 Turbo C 的使用初步。

C 语言同时具有高级语言和汇编语言的特点，是一种功能很强、也很灵活的编程语言，既适合于作为系统描述语言，又适合于作为通用的程序设计语言。

为了区别各个变量、函数、标签和其他用户定义的对象，都必须为它们取不同的名字。这些名字称为标识符。C 语言规定，标识符的第一个字符必须是字母或下划线。后面的字符可以是字母、数字或者下划线。

所有的语言都保留了一些单词供其内部使用，它们被称为关键字。C 语言有 32 个关键字。

C 语言程序由函数构成，一个 C 语言程序总是从主函数(main)开始执行，而不论其在程序中的位置，主函数执行完毕亦即程序执行完毕。

习 题

1-1 编写一个 C 语言程序，在 DOS 屏幕输出以下字符串：

This is my first C!

1-2 请根据自己的认识，写出 C 语言的主要特点。

1-3 写出一个 C 语言程序的构成。

1-4 C 语言用途是什么？它和其他高级语言有什么异同？

1-5 上机运行本章所有例题，熟悉上机的方法与步骤。

第 2 章 数据类型与表达式

学习目标

- 1) 了解常量和变量的概念。
- 2) 掌握各种数据类型及其类型说明, 其中涉及到的重要概念有: 整型、实型、字符型数据的表示、存储、取值范围、数值有效位数及各种类型说明形式。
- 3) 掌握各种运算符与表达式, 其中涉及到的重要概念有运算对象的个数、运算优先级、结合性、类型转换等。

2.1 数据与数据类型

数据是程序加工、处理的对象, 也是加工的结果, 所以数据是程序设计中所要涉及和描述的主要内容。程序所能够处理的基本数据对象被划分成一些组, 或者说是一些集合。属于同一集合的各数据对象都具有相同的性质。例如, 它们能够做同样的操作, 它们都采用同样的编码方式等, 把程序语言中具有这样性质的数据集合称为数据类型。计算机基础知识告诉我们, 计算机硬件也把被处理的数据分成一些类型, 有定点数、浮点数等。CPU 对不同的数据类型提供了不同的操作指令, 程序语言中把数据划分成不同类型与此有密切关系。但在程序语言中, 类型的意义还不仅于此。所有程序语言中都是用数据类型来描述程序中的数据结构、数据表示范围、数据在内存中的存储分配等。实际上数据类型是计算机领域中一个非常重要的概念, 可以说是计算机科学的核心概念之一。在学习程序设计的过程中, 我们将要不断地与数据类型打交道。

在 C 语言中, 任何数据对用户呈现的形式有两种: 常量或变量。而无论常量还是变量, 都必须属于不同的数据类型。在一个具体的程序设计语言系统里, 每种数据类型都有固定的表示方式, 这个表示方式实际上就确定了可能表示的数据范围和它在内存中的存放形式。例如, 一个整数类型就是数学中的一个整数子集, 其中只能包含有限个整数, 超出这个子集之外的整数在这个类型里是没有办法表示的。

C 语言提供的主要数据类型具体分类如下:

- 1) 基本类型。分为整型、实型(又称浮点型)、字符型和枚举型四种。
- 2) 构造类型。分为数组类型、结构体类型和共用体类型三种。
- 3) 指针类型。
- 4) 空类型(无值类型)。

C 语言为每个类型定义了一个类型名, 用来区分不同的类型, 如整数类型的名称是 `int`, 字符类型的名称是 `char` 等。

C 语言的数据类型比其他一些程序设计语言的数据类型要丰富, 它有指针类型, 还有构造其他多种数据类型的的能力, 构造类型一般是由基本数据类型按照一定的规则构造而成, 结

构相对基本类型来说比较复杂。本章主要介绍基本数据类型，其他数据类型将在本书以后的章节中介绍。

2.2 C 语言中的基本数据类型

2.2.1 常量与变量

1. 常量

在程序运行过程中，其值不可以被改变的量称为常量。常量分为不同的类型，在 C 语言程序中，常量是直接以自身的存在形式体现其值和类型的，如 12、-3 是整型常量，1.5 是实型常量。

在 C 语言程序中，常量除了以自身的存在形式直接表示之外，还可以用标识符来表示常量，称为符号常量。

C 语言中用宏定义来对符号常量进行定义，其定义形式如下：

```
#define 标识符 常量
```

其中 #define 是宏定义命令的专用定义符，标识符是对常量的命名，就是符号常量。例如：

```
#define PI 3.14159
```

其中 PI 为一个符号常量，C 编译系统在处理程序时会将程序中全部的 PI 均用 3.14159 代替。

例 2-1 已知圆的半径 r ，求圆的周长 c 和圆的面积 s 。

```
#define PI 3.14159
main()
{ float r,c,s;
  scanf("%f",&r);          /* 输入圆的半径 */
  c = 2 * PI * r;
  s = PI * r * r;
  printf("c = %f\n,s = %f\n",c,s);
}
```

在使用符号常量时要注意，符号常量不同于变量，其值在它的作用域内是不能被改变的，也不能够重新赋值。为了区别于变量，通常符号常量名用大写字母拼写，而变量名则用小写字母拼写。

2. 变量

在程序运行过程中，其值可以被改变的量称为变量，这里所说的变量与数学中的变量是完全不同的概念。在 C 语言中，变量是用来保存数据的，它有确定的值，只是在不同的时候有不同的值。

程序里的一个变量可以看成是一个存储数据的容器，它的功能就是可以存储数据。对变量的基本操作有两个：一是向变量中存入数据值，这个操作被称作给变量“赋值”；二是取得变量当前值，以便在程序运行过程中使用，这个操作被称为“取值”。