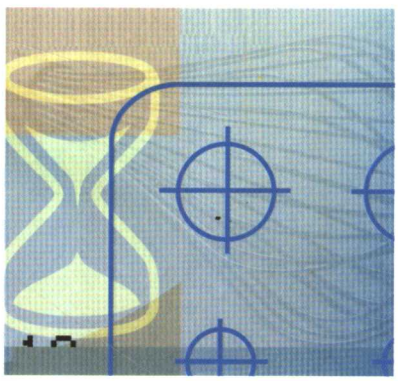
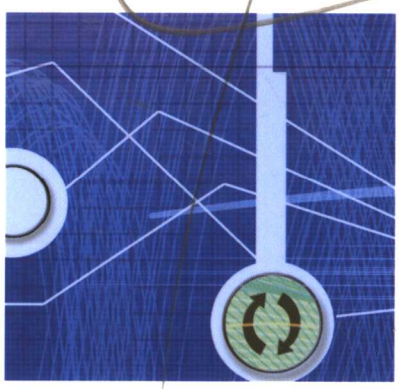
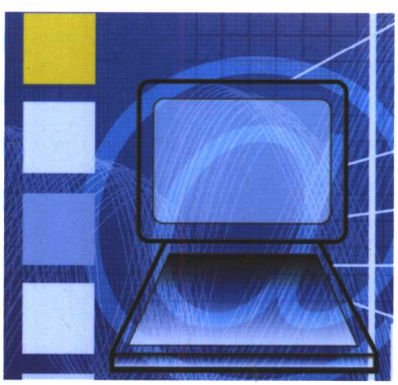
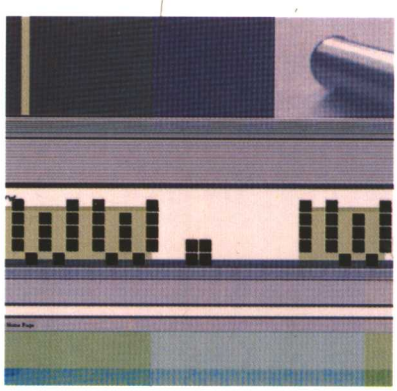
21世纪高等学校应用型教材

C程序设计教程

□ 谭浩强　主　编
□ 卜家岐　范燮昌　编著

高等教育出版社
Higher Education Press

21 世纪高等学校应用型教材

C 程序设计教程

谭浩强　主编

卜家岐　范燮昌　编著

高等教育出版社

内容提要

本书介绍C语言的相关知识，共10章，主要内容包括：C语言概述；数据类型、常量、变量和数组；运算符、表达式和常用的输入/输出函数；顺序结构和选择结构；循环结构和无条件转向语句；数组的应用；函数；结构体、共用体和枚举类型；文件；综合应用。本书配有辅导书《C程序设计教程上机辅导与习题集》。

本书适合作为大学本科、高职高专、成人高校和其他初学者学习C程序设计的教材，也可供参加全国计算机等级考试（二级C）的各类读者选用。

本书所配电子教案及书中案例程序设计源代码均可以从高等教育出版社高等理工教学资源网站下载，网址为：http//www.hep-st.com.cn。

图书在版编目(CIP)数据

C程序设计教程 / 谭浩强主编；卜家岐，范燮昌编著.
—北京：高等教育出版社，2006.1
ISBN 7-04-018143-6

Ⅰ.C… Ⅱ.①谭… ②卜… ③范… Ⅲ.C语言-程序设计-教材 Ⅳ.TP312

中国版本图书馆CIP数据核字（2005）第143562号

策划编辑 雷顺加　**责任编辑** 萧　潇　**封面设计** 王凌波　**责任绘图** 朱　静
版式设计 胡志萍　**责任校对** 朱惠芳　**责任印制** 宋克学

出版发行	高等教育出版社	**购书热线**	010-58581118
社　　址	北京市西城区德外大街4号	**免费咨询**	800-810-0598
邮政编码	100011	**网　　址**	http://www.hep.edu.cn
总　　机	010-58581000		http://www.hep.com.cn
		网上订购	http://www.landraco.com
经　　销	蓝色畅想图书发行有限公司		http://www.landraco.com.cn
印　　刷	北京凌奇印刷有限责任公司	**畅想教育**	http://www.widedu.com
开　　本	787×1092　1/16	**版　　次**	2006年1月第1版
印　　张	16.75	**印　　次**	2006年1月第1次印刷
字　　数	400 000	**定　　价**	21.20元

物料号　18143-00

前　言

C 语言是C++语言的基础，适用于程序设计的初学者。用 C 语言可编写风格优美的应用程序，又能编写计算机的系统软件。由于其表达简洁，功能丰富，使用灵活，应用广泛，目前高校中许多专业都开设 C 语言相关课程，甚至把“C 语言程序设计”作为必修基础课。

本书力求通俗易懂，入门容易，重视概念，加强实践。全书包含 128 个程序设计实例（其中包括 2 个综合练习题和 3 个综合应用题），使读者通过实践掌握 C 语言的基础知识。为使读者更好地掌握 C 语言中的难点“指针”，本书将其分散教学，指针的概念从第 1 章就开始引入并贯穿全书，使初学者在学习过程中循序渐进，逐步深入，反复实践，牢固掌握指针的应用。

本书共分 10 章，第 1 章概括介绍 C 语言，第 2 章介绍数据类型、常量、变量和数组，第 3 章介绍运算符、表达式和常用的输入/输出函数，第 4 章介绍顺序结构和选择结构，第 5 章介绍循环结构和无条件转向语句，第 6 章介绍数组的应用，第 7 章介绍函数，第 8 章介绍结构体、共用体和枚举类型，第 9 章介绍文件，第 10 章介绍综合应用。

本书内容以 ANSI C（美国国家标准 C 语言部分）为基础，应用传统流程图和 N-S 结构化流程图描述算法，程序写成锯齿形缩进格式，这些都有助于读者养成良好的编程习惯，编写出可读性好，质量高的应用程序。

全书例题全部通过 Turbo C 2.0 集成开发环境调试通过，当然也可用 Visual C++6.0 集成开发环境来调试和编译这些程序（详见配套用书《C 程序设计教程上机辅导和习题集》第 2 章“用 Visual C++6.0 开发 C 程序”）。前者简单易学，但只能用键盘，不能用鼠标；后者功能强大，可使用鼠标，而且字符串和注释可用中文，但对初学者有一定的难度。读者可根据自己的具体情况选用。

本书由全国高校计算机基础教学研究会理事长谭浩强教授主编，主要由上海大学卜家岐教授和范燮昌副教授执笔编写，参加本书大纲讨论、程序调试及部分编写工作的还有：陈章进、侯庆祥、康惠骏、邹启明、王晓冬、罗真、徐安东、陆黎明、黄俊民、余慧蔼等。作者都具有丰富的 C 语言教学经验和软件设计的实践。初稿完成后又请了三位多年从事 C 语言一线教学的主讲教师（北京科技大学的徐新华副教授、同济大学高等技术学院的程克明副教授和承德石油高等专科学校的马晓晨副教授）进行了认真的审读，他们提出了不少宝贵意见，根据审读者所提意见，作者对原稿又作了认真的研究、讨论和修改，使本书的实用性进一步加强，在此对他们表示真挚的谢意。

由于作者水平有限，书中肯定会有缺点和错误，诚请读者指正。

作者

2005 年 10 月

目　录

第 1 章

C 语言概述

本章要点

- 了解 C 语言在开发计算机应用软件和系统软件中的重要作用。
- 理解冯·诺依曼计算机的概念。
- 建立字节、内存单元、地址和指针的概念；掌握变量读/写的两种方式。
- 了解结构化程序中三种基本结构的概念。
- 掌握 C 程序的组成和函数的概念。
- 上机熟悉开发 C 程序的集成环境，区分源程序、目标程序和可执行程序的关系。

1.1 计算机和 C 语言

现在，计算机除了能完成极其复杂的数学计算工作外，还成为一种现代化的处理信息的工具。文字信息、图像信息、视频信息和音频信息都可以通过计算机进行存储和处理。现代计算机技术的发展依赖于组成计算机的两大系统——硬件系统质量的提高和软件系统的大量开发。计算机硬件系统的质量提高很大程度上取决于 CPU（Central Processing Unit，中央处理单元）运算速度的提高和内存储器容量的不断扩大。

计算机的软件系统的开发指的是程序的开发。计算机做任何工作，都是通过存储在内存中的程序来实现的，可以说程序是计算机的“灵魂”。C 语言在程序开发设计中起到极其重要的作用，是编制系统软件和应用软件的重要工具。

C 语言最初是作为编写系统软件——包括操作系统（Operating System，OS）、编译程序（Compiler）、解释程序（Interpreter）以及一些集成（多功能）开发环境等——的程序设计语言而发展起来的。UNIX 操作系统就是用 C 语言编写的一个极其成功的例子。随着在各个领域中的应用不断广泛和深入，C 语言在很多方面显示出不同于其他程序设计语言的特色和优点，吸引了越来越多的使用者和研究者，这又进一步促进了 C 语言的发展。

要全面掌握 C 语言的特点，首先应从认识二进制和程序存储控制原理开始，循序渐进逐步掌握 C 语言的丰富内涵。

1.2 二进制和程序存储控制原理

二进制和程序存储控制原理是由美籍匈牙利裔数学家冯·诺依曼（Von Neuman）提出的。

他认为，如果用二进制而不是十进制表示数值，就可以利用电路的开和关两种状态来表示二进制中的0和1，从而利用电子文件进行数值运算。在此理论基础上，1946年诞生了世界上第一台电子计算机ENIAC（Electronic Numerical Integrator And Computer）。它用二进制代替十进制完成了复杂的数值运算，但是运算指令和数据是通过重新连接电路和设定开关来完成的。通过实践，冯·诺依曼又提出存储程序（Stored Program）的概念，即将运算指令和数据用数字（二进制）的方式存放在电子计算机的存储器中，即用程序来控制计算机的操作。这就是著名的**冯·诺依曼原理**——二进制和程序存储控制的计算机结构思想。现在实际应用的计算机绝大多数都基于冯·诺依曼原理，因此称为**冯·诺依曼计算机**。

在主机内的存储器，称为内存储器，简称**内存**。计算机要运行的程序和数据都存放在内存中。那么内存是由什么组成的呢？它是由上千万个具有两个状态的电子开关组成的。电子开关的两个状态代表二进制数的“0”和“1”。每个电子开关称为一个**位**（**bit**），它可以代表二进制数（逢二进一）的一个基本单元。计算机内存就是一个庞大的二进制基本单元——电子开关的集合体。

要存放信息，必须把这些电子开关有机地组织起来。一般用若干个二进制位组成一个**字节**（**byte**）。多数计算机用8位组成一个字节，如图1.1所示。

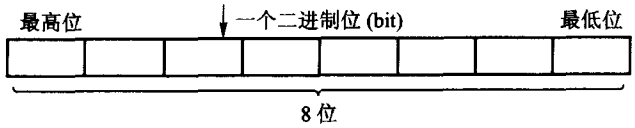


图1.1 字节示意图

一个字节可以存放一个字符，当然这个字符是用二进制数来表示的，这个二进制数就是该字符的二进制代码。计算机上所用的全部字符（字母、数字及其他专用字符）都有相应的二进制代码，如字母“a”，它的二进制代码是：

0	1	1	0	0	0	0	1

很多计算机系统采用ASCII（American Standard Code for Information Interchange美国标准信息交换码）定义字符和二进制代码之间的对应关系（见附录Ⅰ）。

为了管理方便，每个字节在内存中都有相应的位置编号，这个编号就是该字节的**地址**。通过地址可以找到内存中任何一个字节的内容。

一个字节可以存储一个字符，但要存储一个整数或一个实数，一个字节就不够了。有的系统中用2个字节或4个字节来存储一个整数，用4个字节或8个字节来存储一个实数。由一个或若干个字节组成一个**字**，一个字也称为一个**存储单元**（**或内存单元**），可以用来存放一个数据或一条指令。**程序即一系列指令和数据的集合体**。

由于内存中每个字节都有自己的地址，因此每个存储单元也有自己相应的地址。由一个字节组成的存储单元的地址（内存单元的地址）就是这个字节的地址；而由多个字节组成的存储单元的地址（内存单元的地址）指的是组成这个存储单元的几个字节中第一个字节的地址。

假定程序中已定义了三个整型变量a、b、c，编译时系统分配给它们各一个存储单元，每

个存储单元占两个字节，而变量 a、b 和 c 在内存中存放的地址在 C 语言中可用&a、&b 和&c 表示，其中“&”是 C 语言中的**地址运算符**，即通过运算符&可以得到相应变量的地址。所以，当程序定义了某一变量以后，变量名和它占用的存储单元地址有直接的对应关系。如储存的程序要实现如下运算：

```
b=5;
c=8;
a=b+c;
```

则系统会把 5 和 8 分别送到内存地址为&b 和&c 的两个存储单元中存放起来，而后又将地址为&b 和&c 单元中的 b 值和 c 值取出，通过 CPU 将它们相加后得和 13，送到变量 a 的单元（即地址为&a 的存储单元）中。这种按变量所对应的地址存取变量的方式称为**直接访问**方式，“访问”即在存储单元中存取（或称读/写）数据。直接访问方式使用起来比较容易，比较直观。

在这里要区别存储单元的地址和存储单元的内容（即变量值）这两个概念。对变量值的存取是根据变量名和地址的直接对应关系来进行的，根据变量名和地址的对应关系，先找到变量的地址，然后从这个地址所标识的存储单元中进行存取数据的操作。

通过变量的地址能找到变量的存储单元，因此，C 语言中把变量的地址也称为变量的**指针**。假如&b 的值为 5000，则 5000 就是变量 b 的指针。若内存中变量是按顺序存放的，那么地址 5002 应该是变量 c 的指针。通过变量的指针可以找到变量的存储单元来对变量进行读/写操作。指针指的是某个存储单元中第一个字节的地址（首字节地址或说一个存储单元的入口地址）。

与直接访问相对应的另一种访问内存的方式是**间接访问**方式。C 语言中可以定义一种变量专门用于存放地址。假定我们定义一个变量 x_pointer，专门用来存放整型变量地址，系统编译时也分配给这个变量一个存储单元，假定它的地址是 8620，那么可以通过下面的赋值语句将变量 b 的地址赋给变量 x_pointer：

```
x_pointer=&b;
```

变量 b 的地址值是 5000，所以通过以上赋值语句就把地址值 5000 放入地址以 8620 开始的两个字节中，x_pointer 的值就是 5000。这样，我们就说 x_pointer 指向变量 b，可以对变量 b 进行间接访问了，过程是：先从变量 x_pointer 中提取变量 b 的指针 5000，然后到地址是 5000 和 5001 两个字节中存取 b 的值。这个过程，C 语言中可以用一个存取内容运算符“*”来运算，即用*x_pointer 来间接存取 b 的值。如

```
*x_pointer=5;          /*间接访问“存”*/
```

等效于

```
b=5;                   /*直接访问*/
```

这两个语句的结果都是对变量 b 赋值 5，但它们执行的过程不同，一个是通过间接访问，而另外一个是直接访问。因此

```
a=*x_pointer+c;        /*b 间接访问“取”，c 直接访问*/
```

等效于

```
a=b+c;                 /*b、c 都是直接访问*/
```

当 x_pointer 的值用变量 c 的地址&c 重新赋值：

```
x_pointer=&c;
```

那么我们说 x_pointer 指向变量 c（这时 x_pointer 已经脱离了与变量 b 的联系），借助于 x_pointer 可以对变量 c 进行读写操作：

```
*x_pointer=8;
```

等效于

```
c=8;
```

因为 x_pointer 的值可以改变，所以称为**指针变量**。通过指针变量来存取某变量的方式称为**间接访问**的方式。关于指针变量的定义和应用将在以后各章逐步展开，这里只做概述，目的是借助它说明间接访问内存的方式。

可见，计算机内存中数据的传递和存取都是二进制的，程序以一系列指令和数据的形式存放在一个个存储单元中，执行程序的过程也是不断访问存储单元的过程，而访问存储单元的方式可以是直接访问，也可以用间接访问。这就是冯·诺依曼提出的二进制和程序存储控制原理。这个概念很重要，是学好以后各章的基础。

1.3 整数在内存中的存放形式

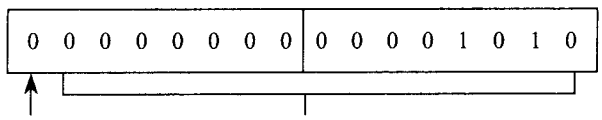
内存中的每一个数都占用一个存储单元，并以补码的形式存放。关于补码的知识虽不属于本课程的范围，但对学习 C 语言很重要，这里以整数为例做简单介绍。

一般整数占 2 个内存字节，这两个字节中的最高位用来表征数的符号，以 0 代表正数，以 1 代表负数，其余各位代表数的绝对值。

一个整数的代码有原码、反码和补码三种形式。**正数的原码、反码和补码的形式相同**。例如，整数 10 在内存中占两个字节，而+10 的原码、反码和补码都具有如下的形式：

```
0000000000001010
```

写得更清楚一些，用 2 个字节表示如下：



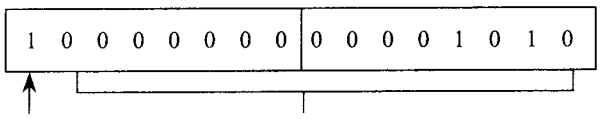
符号位　　数字位（数的绝对值）

0 代表正

负数的原码、反码和补码的表示形式是不同的。如–10 的原码是：

```
1000000000001010
```

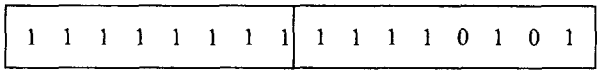
写得更清楚一些，用 2 个字节表示如下：



符号位　　数字位（数的绝对值）

1 代表负

负数的反码规定：**符号位不动，其余各位对原码取反**。如–10 的反码是：



负数的补码规定：**一个负数的补码是它的反码加 1**。如–10 的补码是：

用补码的好处是系统在进行运算时，数的符号位用不着单独处理，同时数字 0 是唯一的。

当已知一个负数的补码反过来要求其原码时，只要记住“**一个负数补码的补码就是它的原码**”，即把补码当原码，再求一次补码就得原码。如以–1 为例：

原码： 1000000000000001
反码： 1111111111111110
补码： 1111111111111111 （内存中的存储形式）
补码取反加 1 得原码： 1000000000000001

1.4 C 程序设计语言概述

1.4.1 程序设计语言的发展

程序是一系列指令和数据的集合体。那么我们怎样来设计这个集合体呢？早先，人们只知道计算机内部的指令和数据都是以二进制形式存储的，计算机内部传递信息并实现各种操作也是由二进制的指令来完成的。也就是说，计算机内部只能识别和执行二进制代码。因此，在计算机发展初期，人们就是用手工编写出一条条二进制指令，这些二进制指令的集合就称为**机器语言**。这种机器语言都是由 0 和 1 组成，实际使用起来非常繁琐，极易出错。

后来，人们想到用符号来代替二进制形式的指令，符号指令便于记忆和使用。这些符号也称作“助记符”。例如，用 ADD 代表两数相加，用 MOV 表示数据传送等。由这些符号组成的指令集称为“符号语言”，也称“**汇编语言**”。计算机不能直接接受汇编语言，要通过“翻译”，把汇编语言翻译成计算机能接受的二进制的机器语言，这个翻译工作是通过一个事先装入计算机中的“汇编程序”软件来完成的。汇编语言比机器语言简单易学，为专业人员设计系统软件提供了很大的方便。但对于一些非专业人员来说，仍感到难以掌握，而且不同的机器都有自己的一套编码，这就严重地影响了计算机的进一步推广应用。

直到 20 世纪 50 年代，世界上出现了第一种“高级语言”——FORTRAN 语言，从此计算机的应用得到了飞速的发展。

高级语言更接近于人们习惯的自然语言和所用的数学公式。如要计算机打印输出只要输入“print”；要打开某文件的通道输入“open”；要计算 x 的余弦函数用 cos(x)；等等。这为非计算机专业的人员应用计算机提供了极大的方便。继 FORTRAN 语言以后，高级语言发展迅速，至今已有几百种不同的高级语言，其中 C 语言就是一个突出的代表，它适用于多种领域，特别适用于编写系统软件和工程应用软件等。

C 语言简洁、方便、灵活，且具有各种控制流结构，以满足结构化程序设计的需要。C 语言的代码质量高，程序运行速度快。

在C语言的基础上1983年首次推出C++语言，C++语言是C语言的超集，它继承了C语言的全部优点，并提出程序设计的新概念和新方法，自20世纪90年代以来开始成为程序设计的新潮流。

1.4.2 结构化程序设计方法

程序设计的方法很多，程序员可以充分发挥自己的聪明才智，设计出形式多样、运行效率很高的程序。但是，对于一个效率很高的程序，如果别人在阅读或调试时需费九牛二虎之力，软件日后的维护又非常困难，这样的程序仍不是一个理想的程序。

荷兰学者E.W.Dijkctra提出**结构化程序设计**（Structured Programming，SP）的理论。经过多年的实践，结构化程序设计的理论和实施方法日益完善并已被大家承认和接受。结构化程序设计方法提出了编程时必须遵守的一些原则，目的在于保证程序的可读性，便于推广应用和交流。为了方便记忆，可以把这个原则归纳为32个字：**“自顶向下，逐步细化；清晰第一，效率第二；书写规范，缩进格式；基本结构，组合而成”**。

1．自顶向下，逐步细化

这一原则是指将任务或问题的总要求划分为若干个相对独立的模块，而每个模块根据它所完成的功能又可细分成若干个子模块，如此从上而下，逐步细化，一直细化到不能再分割的模块为止。然后再考虑每个模块中使用的具体函数和语句。

2．清晰第一，效率第二

这一原则是从提高程序的可读性，方便交流、调试、修改和维护的角度出发。程序只有在结构清晰的基础上，才去考虑它的效率。

3．书写规范，缩进格式

这是程序设计风格的重要要求之一。写程序时不要把语句“堆”在一起，而是将程序的不同层次逐行向右缩进，写成锯齿形。主函数main()的函数体（一对大括号中的语句）应向右缩进几列来书写。若函数体中还有其他模块（如以后要学习的条件转移模块和循环模块等），这些模块内的语句在主函数体向右缩进的基础上还要再往右缩进几列，依此类推，即凡是模块内的模块都要求进一步地向右缩进，形成锯齿形程序格式。这样的程序清晰易读，纠错容易。具体例子可参见本书各章的例题。读者在平时的练习中要养成书写程序的良好习惯。

4．基本结构，组合而成

根据结构化程序的基本要求，不论程序大小，简单还是复杂，规定程序主要由三种基本结构组成，即顺序结构、选择结构和循环结构。实践证明，用这三种基本结构可以组成各种各样的程序模块，实现任何复杂的算法。

C语言完全支持结构化程序设计，它有多种结构化语句，如if语句、while语句、do_while语句以及for语句等，供用户在不同场合选用。

1.4.3 C语言简史

C语言是怎样发展而来的呢？简单地说，C语言是由B语言发展而来，而B语言又由A语言发展而来。

A语言指的是高级语言Algol 60。1960年出现的Algol 60是一种面向过程的高级语言，它离硬件远，不适合用来编写系统软件。

1967年英国剑桥大学的Matin Richards在Algol 60基础上推出了比较接近硬件的BCPL（Basic Combined Programming Language）语言，1970年美国贝尔实验室的Ken Thompson以BCPL语言为基础又设计出简单而又很接近硬件的B语言，并用B语言写了UNIX操作系统初版。

1973年贝尔实验室的Dennis M.Ritchie在B语言的基础上又设计出C语言，它保留了B语言精炼和接近硬件的优点，又克服了它过于简单、无数据类型等缺点。后来Ken Thompson和Dennis M.Ritchie合作，进一步改写UNIX操作系统。到1977年UNIX操作系统得到日益广泛使用，与此同时C语言也迅速得到推广。1978年以后，C语言已先后移植到大、中、小、微型机上，C语言已风靡全世界。

1978年Brian W.Kernighan和Dennis M.Ritchie合著了《The C Programming Language》一书，这本书成为后来广泛使用C语言的基础，称为标准C。后来，美国国家标准协会ANSI（American National Standard Institute）又推出了新的标准C，它比原来的标准C又有了较大的发展，1988年Brian W. Kernighan和Dennis M. Ritchie重写了他们的经典著作《The C Programming Language》。目前我国很多院校仍使用该书作为教材。

1980年贝尔实验室的Bjarne Stroustrup开始对C语言进行改进和扩充，1983年正式取名为C++（使用了C语言中的自增运算符++，表示是C语言的进一步扩充和改进），以后又经几次修订，于1994年制订了ANSI C++标准草案。目前，C++仍在不断发展中。由于C++的出现，有人称原来的C语言为“传统的C语言”。

1.4.4 C语言的字符集和标识符

1. C语言的字符集

C程序中的单词或标识符由下列字符组成：

① 小写英文字母：a～z

② 大写英文字母：A～Z

③ 数字字符：0～9

④ 27个特殊字符：+ – * / = : ; ? \ ~ | ! # % & ()
[] {} ^ < > _（下划线） 空格 , . " '

请注意，程序中不可以包含除以上四种字符之外的其他字符。

2. C语言中的标识符

标识符是程序设计人员用来命名程序中的一些基本单元或模块，如类型名、对象名、函数名、变量名、常量名等。C语言规定：**标识符**由字母、数字字符和下划线组成，并以字母或下划线开头。

定义标识符时应遵守下列约定：

① 定义标识符时，避免使用系统已经定义过的标识符。这些标识符对C语言的编译程序来说有着特殊的含义，是系统的保留字，也称关键字（见附录B）。

② 标识符中的大小写字母是有区别的。如A和a、Print和print、Year和year等都代表不

同的标识符。

③ 实际编程时，提倡使用有意义的英文单词，最好能做到“见名知义”，如求和变量可以用 s 或 sum；阶乘变量可以用 fa 或 factorial；医生工资可以用 dp 或 doctorPay；工人年龄可以用 wa 或 workerAge 等，帮助提高程序的可读性。

④ 标识符的长度（字符个数）可以是任意的，不同的编译系统识别标识符的长度不一样，一般至少能识别 8 个字符，有的编译系统最多能识别 32 个字符。

1.4.5 C 程序的基本结构

1. 一个简单的 C 程序

例 1.1 求两数乘积

```
/* ex1_1.c the value of b*c */
#include "stdio.h"
main()
{
    int a,b,c;
    printf("Enter two int numbers:\n");
    scanf("%d%d",&b,&c);
    a=b*c;
    printf("a=%d\n",a);
}
```

运行结果：

```
Enter two int numbers:
5   8   ↙(本书中符号“↙”表示按 Enter 键)
a=40
```

2. C 程序的组成

上面这个程序很简单，但已经可以看出，C 程序是由函数、语句、注释和编译预处理命令等内容组成的。下面分别加以说明。

（1）函数

函数是程序设计模块化的体现。函数用来完成某个特定的操作，如例 1.1 中的 main()函数、scanf()函数和 printf()函数。后面两个函数是被调用函数，它们包含在头文件 stdio.h 中（见编译预处理命令）。一个程序可以包含很多函数。这些函数可以是由用户自己设计的（见第 7～10 章），也可以是系统提供的，通过编译预处理命令把它们包含进来，如本程序的 scanf()函数和 printf()函数。但一个程序中一定要有一个主函数 main()，并只允许有一个，主函数可以放在程序的任何地方（可以放在众多函数的最前面、中间的某个地方或最后），但不管主函数放在程序的哪个部位，程序的执行都是从主函数开始，更明确地说，程序的执行就是执行主函数，其他函数只能通过主函数或被主函数已经调用的函数调用。

（2）语句

C 程序中的每个函数主要由语句组成，从例 1.1 中可以看到，一个语句的结束标志是分号。如：

```
int a,b,c;                                    /*变量说明语句*/
```

```
printf("Enter two int number:\n");        /*输出提示信息语句*/
scanf("%d%d",&b,&c);                      /*输入语句*/
a=b*c;                                    /*表达式语句(或赋值语句)*/
printf("a=%d\n",a);                       /*输出语句*/
```

表达式语句（或赋值语句）是C程序中应用最多的语句。其他还有通过数据类型定义变量或数组，后面也带分号，则可称为变量或数组的说明语句。只有一个分号而前面没有内容的语句称为空语句等。

另外还有输入语句、输出语句、选择结构语句、循环结构语句和无条件控制语句等，这在以后章节中会详细讲解。

（3）注释部分

C程序中为了说明程序的功能或某几行的含义，可带注释。注释能帮助读者阅读和理解程序。程序编译时，注释被忽略，它不产生代码行。放在程序开头的注释，说明整个程序的功能，放在其他部分的注释，说明局部程序或语句的功能。例1.1中程序开头用了如下注释：

```
/*   ex1_1.c   the value of b*c */
```

说明程序用来求b乘以c的值。注释内容写在一对符号“/*”和“*/”之间，这是传统C中的注释方式，其中的内容可以是一行或几行。自符号“/*”开始到“*/”符号结束，其间的内容都被认为是注释内容。

（4）编译预处理命令

C程序的开头一般可以看到有些程序行以“#”符号开头，这些程序行就是编译预处理命令。C提供三类编译预处理命令：文件包含、宏定义和条件编译（见第7章）。例1.1中有文件包含命令：

```
#include  "stdio.h"
```

其中stdio.h是一个头文件，也称标准的输入/输出头文件。程序中由于要用到数据的输入函数scanf()和输出函数printf()，而这两个函数的定义和说明，系统已经存放在头文件stdio.h中，因此要包含该头文件。所谓包含就是把头文件代码引入程序中，由于这个工作是在编译程序前完成的，所以称编译预处理命令。

函数、语句、注释和编译预处理命令是构成程序的四个基本成分。

1.4.6 C程序的编辑、编译和连接

用计算机高级语言编写的程序，称**源程序文件**。源程序通过编译程序（也称编译器）编译以后生成二进制的目标代码，即机器语言指令，也称为**目标程序文件**（简称**目标文件**），通常以.obj作为文件的扩展名。

通常C程序中可能包含几个目标文件或库文件（扩展名为.lib），它们都是二进制文件，应把它们连接起来，生成一个完整的可执行文件，这个连接工作由专门的软件——连接程序（又称连接器）来完成。连接以后，系统生成可执行程序文件（简称可执行文件），文件的扩展名是.EXE，所以也称EXE文件。源程序通过编译和连接后产生一个可执行文件，这个过程一般来说不可能一次成功，有一个反复调试的过程。

从编辑源程序开始一直到产生一个可执行文件，一般都在一个集成开发环境（Integral

Development Environment，IDE）中完成，这个开发环境集编辑、编译和连接于一体，在编译与连接过程中，根据是否有出错信息来决定是否要重新编辑或修改源程序。图 1.2 所示为产生一个 C 语言可执行文件的过程。

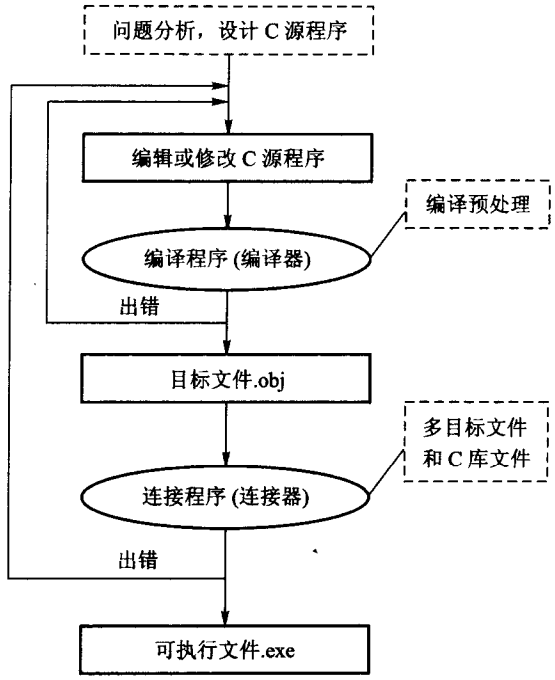


图 1.2 产生 C 语言可执行文件的过程

对初学者来说，目前应用较多的集成开发环境是 Turbo C 2.0。它简单易学，容易入门；缺点是只能用键盘，不能用鼠标来操作。为此，在本书配套的《C 程序设计教程上机辅导和习题集》中除详细介绍 Turbo C 2.0 集成开发环境外，还增加了一章，介绍在 Visual C++ 6.0 集成开发环境中使用鼠标操作开发 C 程序。Visual C++ 6.0 功能强大，而且字符串和注释可用中文，但该环境对初学者有一定的难度。读者可根据自己的具体情况选用。

为照顾没有辅导书的读者，1.4.7 小节将对 Turbo C 2.0 集成开发环境做一简单介绍。如要详细了解 Turbo C 2.0 和用 Visual C++ 6.0，请参阅《C 程序设计教程上机辅导和习题集》一书。

1.4.7 Turbo C 2.0 集成开发环境简介

进入 Turbo C 2.0 集成开发环境（以下简称 TC）后，屏幕如图 1.3 所示，上面的窗口称为编辑（Edit）窗口，用于编辑源程序。按一下功能键 F6，屏幕切换到信息（Message）窗口或监测（Watch）窗口。信息窗口提供各种调试信息或用来监测变量的变化（此时窗口转换成监测窗口）。编辑窗口下面是状态栏。

状态栏中各功能键的作用如下：

F1：激发帮助窗口。

F2：保存文件。