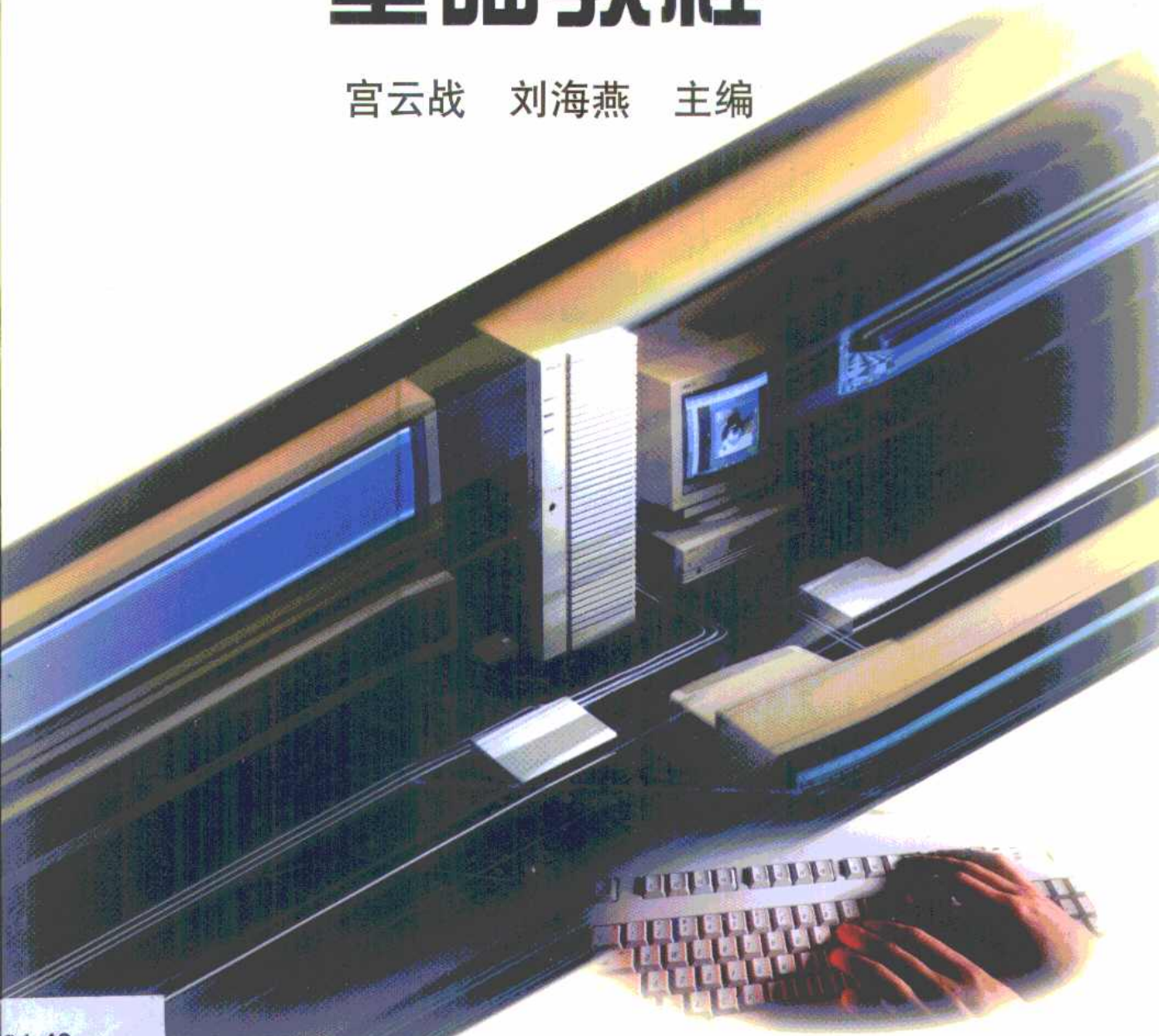
普通高等教育“九五”国家级重点教材

计算机软件技术基础教程

宫云战　刘海燕　主编



机械工业出版社
China Machine Press

14

普通高等教育“九五”国家级重点教材

计算机软件技术基础教程

宫云战　刘海燕　主编

机械工业出版社

本书全面介绍了计算机软件的基础知识。通过本书的学习，可使读者掌握如何编写程序、如何编写一个“好”程序以及如何编写一个大型程序的方法。全书共分 6 章，内容包括：C 语言的基础知识；C 语言程序设计的基本控制结构；C 语言的构造数据类型；C 语言函数和文件的使用；数据结构的基础知识；软件工程的基础知识。为适应教学和便于读者自学，书中配有大量的例题和习题。本书是在讲清基本概念、重点面向使用的原则下编纂而成的，选材注重科学性、先进性和实用性。

本书是“九五”国家级重点计算机系列教材的第四册，是学习计算机软件基础知识的入门教材，可作为高等工科院校非计算机专业的专科教材，也可作为计算机爱好者学习计算机软件基础知识的参考书。

图书在版编目（CIP）数据

计算机软件技术基础教程/宫云战，刘海燕主编.—北京：机械工业出版社，2000.1

普通高等教育“九五”国家级重点教材

ISBN 7-111-07531-5

Ⅰ.计… Ⅱ.①宫…②刘… Ⅲ.软件-高等学校-教材 Ⅳ.TP31

中国版本图书馆 CIP 数据核字（1999）第 50273 号

机械工业出版社出版（北京市百万庄大街 22 号 邮政编码：100037）

责任编辑：周 娟 王 龙 版式设计：武 江

封面设计：姚 毅 责任印制：何全君

北京机工印刷厂印刷·新华书店北京发行所发行

2000 年 1 月第 1 版第 1 次印刷

787mm×1092mm 1/16 ·9 印张·字数 209 千字

印数 0001—4000 册

定价：14.00 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）68993821、68326677-2527

前　言

目前，计算机的应用已深入到社会的各个领域，成为人类文化的重要组成部分。计算机已成为当代知识分子分析问题、解决问题的重要工具，应用计算机的能力是现代人高科技素质的重要标志之一。

在我国高等院校，非计算机专业的计算机层次教育（即计算机文化基础、计算机技术基础、计算机应用基础）已逐步被社会所认同。近年来，随着 PC486 及 Pentium 系列等高档微型机的出现，“Windows 文化”在微型计算机中的统治地位日趋加强，计算机的使用已从要求会使用 DOS 及 DOS 系列软件变成要求会使用 Windows 系列软件。在此背景下，编写一套适合高等院校使用的计算机系列教材已非常必要，这套教材已被教育部列为普通高等教育“九五”国家级重点教材。

《计算机软件技术基础教程》是此系列教材的第四册，它属于计算机技术基础的范畴。本书共分 6 章：第 1～4 章介绍了 C 语言的基础知识和 C 程序设计，包括 C 语言的基本概念、控制结构、构造类型数据及函数和文件的使用等，通过这 4 章的学习可使读者学会如何编写计算机程序。第 5 章是数据结构的基础知识，介绍了数据结构的基本概念、线性表、栈与队列、二叉树、图、排序与查找的基本概念和基本算法，掌握数据结构就能够编制一个“比较好”的程序。第 6 章是软件工程的基础知识，同时介绍了目前广为流行的面向对象程序设计的基本概念，通过本章的学习，可使读者掌握大型软件开发的一般方法。书中的附录给出了基于 DOS 和 Windows 的 C/C++程序上机调试的一般步骤。

本书是遵循“讲清基本概念、循序渐进、深入浅出、通俗实用”的原则编纂而成的，内容丰富，编排合理。为便于教学和自学，书中配有大量的例题和习题。本书可作为高等工科院校非计算机专业二、三年级 60 学时的专科教材。

本书在编写过程中承蒙许多专家、教授的热心帮助。国防科技大学陈怀义教授在主审期间，提出了许多建设性意见。国防科技大学齐治昌教授、邹鹏教授，北京大学王立福教授、程旭教授，总参第五十一研究所陈立杰教授，北京航空航天大学金茂忠教授，军械工程学院陈致明教授，装甲兵指挥学院刘湖平副教授，装甲兵工程学院杨醒民教授、周启煌教授等对本套教材的编写工作提出了许多宝贵的建议，谨此向他们表示诚挚的感谢。

由于作者的水平有限，书中一定存在不少不足之处，恳请读者批评指正。

主 编　宫云战

1999 年 12 月于北京

第 1 章　C 语言基础

C 语言以其功能丰富、表达能力强、使用方便灵活、目标程序效率高、可移植性好等特点而著称，是近年来很受人们喜爱的一种程序设计语言。本章将介绍 C 语言的基础知识。

1.1　C 语言概述

1.1.1　C 语言的特点

C 语言的历史可追溯到 1963 年在剑桥大学推出的 CPL(Combined Programming Language)。CPL 语言力图在 ALGOL60 的基础上接近硬件，以通过语言的方法描述硬件。CPL 语言由于规模较大而难于实现。1970 年 Matin Richards 对 CPL 做了简化，产生了 BCPL 语言。1967 年美国贝尔实验室在 BCPL 的基础上进一步做了简化，设计出简单而贴近硬件的 B 语言，并用 B 语言改写了 UNIX 操作系统。为改变 B 语言过于简单和生成的目标代码运行速度慢的缺点，1972～1973 年间，D.M.Ritchie 在 B 语言的基础上设计了 C 语言，1973 年，D.M.Ritchie 和 K.Tompson 合作用 C 语言重写了 UNIX 操作系统。

70 年代末，随着微型计算机的发展，C 语言开始移植到非 UNIX 环境中，并逐步脱离 UNIX 系统成为一种独立的程序设计语言。1988 年美国国家标准协会(ANSI)在继承和发展 D.M.Ritchie 和 K.Tompson 的 C 语言的基础上，制定了 C 语言标准，产生了 ANSI C。与其它高级语言相比，C 语言有以下几个特点：

1) C 语言简洁紧凑，总共只有 32 个关键字和 9 种控制语句。C 语言的程序形式自由，学习起来比较容易。

2) 丰富的运算符及表达式类型的多样化，使程序设计者有较大的主动性，有利于提高程序的可读性和目标代码的质量。同时可以实现在其它高级语言中难以实现的运算，对各种不同类型的程序设计有良好的适应性。

3) 具有丰富的数据结构，C 语言具有五种基本的数据类型和多种构造数据类型，此外用户还可以自己定义数据类型，这使得 C 语言能够实现各种复杂的数据结构。

4) C 语言具有一切必备的结构控制语句，以分立的函数为基础实现程序的结构化，提供由多个源文件构造大型程序以及对各个源文件分别编译的能力。

5) C 语言在具备一般计算机高级语言特点的同时，能直接访问物理地址和进行字位运算，所以能直接对计算机硬件进行操作。也就是说，C 语言能以高级语言的结构和编程环境提供类似于汇编语言那样的系统资源操纵能力及程序的执行效率。程序目标代码执行效率只比汇编语言程序低 10%～20%。

6) C 语言程序可移植性好，大部分代码不改动就可以从一种计算机环境移植到另一种计算机环境中。近年来许多重要的软件产品都是由 C 语言开发的，这主要利用了 C 语言可移植性好等特点。

当然，C 语言本身也有其弱点。一是运算符的优先级较多，不容易记忆。另外由于 C

语言的语法限制不太严格，在增强了程序设计的灵活性的同时，也在一定程度上降低了某些安全性，因此对程序设计人员提出了更高的要求。

1.1.2 C 程序的基本组成及处理过程

1. C 程序的基本组成

下面首先介绍一个相当简单的 C 程序，它包含了 C 程序的基本框架。

【例 1-1】一个简单的 C 程序举例。

```
/*文件名：FIRST.C
  输出 Hello，world ！的 C 程序*/
# include <stdio.h>
main( )
{
  printf ( " Hello，world ！ " ) ;
  return;
 }
```

这是一个 C 语言源程序，这个源程序以一个扩展名为.C 的文件形式存储在系统中。程序中包含了 6 行代码，其作用是要在屏幕上显示信息“Hello，world ！”。程序执行完成后由 return 语句返回操作系统。由这个程序可以看到 C 程序的格式：

预处理指令 ——→ #include <stdio.h>

main 函数 ——→ main( )

{

C 代码块 ——→ /*C 程序的语句*/

}

(1) 包容指令　程序中第一行的# include <stdio.h>不是一条 C 语句，而是包容指令。包容指令是 C 语言的编译指令，编译指令也称为预处理指令。预处理指令不必由分号结束。C 语言编译系统根据包容指令#include 读出尖括号内的文件名，并把该文件与该 C 程序装配在一起。用#include 指令所包容的文件绝大多数都是扩展名为.h 的文件，这些文件称为头文件。

(2) 主函数　所有 C 程序都有一个主函数，在一般系统中主函数为 main( )(在 Windows 系统的程序中主函数为 winmain( ))。main( )函数是程序执行的起点，每个可执行的 C 程序都必须有一个且只能有一个 main( )函数，它可以位于程序的任何位置。一个程序可以有一个或多个函数，除 main( )函数外，其它函数可以是系统提供的标准库函数，也可以是程序员自己编制并命名的函数。

(3) 输入和输出　C 语言的输入和输出由标准库函数实现，C 语言提供的控制台(键盘和屏幕)输入和输出的函数分别是 scanf( )和 printf( )，C 语言程序的输入和输出需包容头文件 stdio.h。

(4) 返回语句　return 语句可使函数返回到调用它的函数。在 main( )函数中，可由 return 语句返回到操作系统。根据 C 语言的习惯，main( )函数中的 return 语句可以省略。

(5) 程序的书写风格　C 语言是一个自由格式的语言，一行可以写多个语句，例如 FIRST.C 程序也可以写成：

```
main( ){ printf( " Hello，world ！ " );}
```

C 语言中空格符、水平制表符和垂直制表符、回车符、换行符、换页符统称为“白”字符，程序中以标点符号和白字符作为单词的分隔。由于 C 语言中多个白字符作为一个白字符看待，所以，FIRST.C 的当前这个版本和原版本是等价的，但显然这个版本比原版本要难读得多。尽管 C 语言的编译系统并不关心程序书写的美观性，但是程序的开发人员必须关心这一点。因为程序不仅是人与计算机、更是人与人之间交换算法思想的中介。

(6) 注释　用“/*”和“*/”括起来的部分为注释信息，它可以增加程序的可读性。

2. C 程序的处理过程

为了在计算机上编写 C 程序，需要一个编辑软件，通过编辑软件在计算机上首先键入源程序，对源程序经过编辑修改后保存在磁盘文件中。

C 程序设计语言是一种编译型语言，C 编译系统以 C 源程序作为输入，通过若干遍处理，将这个源程序翻译成计算机可执行的程序。在 PC 系统中通用的 C 语言处理系统有 Turbo C，C++编译系统有 Inprise 公司的 C++ Builder 和 Microsoft 公司的 Visual C++，这些系统都包含了一个集成的程序开发环境。所谓集成程序开发环境是指一个集编辑、查错、编译以及运行为一体的软件开发环境，用户在这个环境中可以完整地完成程序设计全过程。C 程序设计的全过程是：

C 编辑器 ⟶ C 源程序 (.c 或.cpp) ⟶ 预处理器 ⟶ 经预处理器处理后的 C 源文件 ⟶ C 编译系统 ⟶ C 目标文件(.obj) ⟶ 连接程序 ⟶ 可执行程序(.exe)。

C 源程序在编译前必须通过预处理器的处理。预处理可以看成为 C 编译系统的一部分。在实际编译前的预处理阶段由预处理器分析并处理 C 源程序中以“#”开始的指令。

一个好的编译系统是一个相当“聪明”的系统，当编写的 C 源程序给编译系统去处理时，编译系统会发现其中的语法错误，并以恰当的方式向用户报告。程序中的错误通常被称为“bug”，在一个程序中有 bug 并不是一件可怕的事，重要的是将所有的 bug 找出来，并逐一排除它们。

在接触了第一个程序 FIRST.C 以后，就已经可以开始自己的程序设计实践了。学习程序设计的过程是一个由模仿到创造的过程。学习外语强调“语感”的培养，程序设计语言的学习也需要培养“语感”。培养程序设计语言“语感”的途径就是编写程序、上机查找并排除程序中的 bug，然后在运行中发现程序的不足，再不断地改进程序。

1.2　数据类型和变量

程序由对数据的描述和对数据进行操作及处理的语句组成，C 程序中数据可分为变量和常量两大类。数据必须以适当的方式存放在计算机中，以便由语句进行有效的处理。

1.2.1　程序的词法单位

C 程序中不可再分的单位是词法单位。包括关键字、标识符、常量、操作符和其它分隔符，这些词法单位都是由字符组成的。

1. 关键字

关键字是 C 语言的保留字，在程序中表示固定的含义。【例 1-1】的 C 程序中，程序员不能用关键字 main 作为其它用途。C 语言的关键字均由小写字母组成。ANSI C 规定了 32 个关键字。此外，一些 C 语言版本还增加了其它的关键字。

2. 标识符

标识符是程序员在编制程序时为程序中的变量、函数等所定义的名字。标识符可以短至一个字符或长达 31 个字符(随 C 语言的编译系统而定)。标识符必须以字母或下划线开始，在第一个字符后可以包含字母、数字和下划线。但是 C 语言不提倡程序员定义以下划线开始的标识符，也不要在标识符中包含两个连续的下划线。标识符的大写字母和小写字母作为不同的字符对待，C 语言中的习惯是变量名、函数名使用小写字母，而符号常量全用大写字母。

绝对不能用 C 语言的关键字作为标识符，例如不能用 int 作为变量名，因为 C 语言将 int 作为类型说明的关键字。

下面是一些合法的标识符和非法的标识符的例子：

合法的标识符　　非法的标识符

salary，Salary99，TotalSalary　　99Salary，Salary 99，Salary(99)

建议不要采用诸如 x、a、ABC 等作为标识符。这些标识符尽管符合语言的语法，但含义不明，使程序可读性降低，应当用表示实际含义的单词或缩写作为标识符。

3. 标点符号

标点符号是指程序中的下列符号：# ( ) { } , : ;

标点符号是程序语法的要求，例如 C 程序的每个语句都要以分号“；”结束。标点符号不表示任何实际的操作。

1.2.2 数据类型

数据类型决定了该类型对象的存储表示及对该类型对象能执行的操作(包括这些操作的意义)。C 语言的数据类型分为基本类型、构造类型和指针类型。基本类型包括 char(字符型)、int(整型)、float(浮点型)和 double(双精度型，在 C 语言中 float 和 double 统称为实型)，通过关键字 short、long、signed、unsigned 对以上类型修饰后形成的类型，以及表示“无值”的 void。构造数据类型包括数组、结构、共用体和枚举类型。和其它的程序设计语言相比，C 语言拥有更丰富的数据类型，从而可为数据求得更合理的存储方式。同时，C 语言的数据类型之间的转换比其它语言更为灵活。表 1-1 列出了在 16 位机系统中 C 语言的各种基本数据类型、所占二进制位数及取值范围。

表 1-1　16 位机系统中 C 语言的各种基本数据类型

数据类型	类型描述	位数	取值范围
char	字符	8	-128～127
unsigned char	无符号字符	8	0～255
signed char	有符号字符(=char)	8	-128～127
int	整数	16	-32768～32767
unsigned int	无符号整数	16	0～65535
signed int	有符号整数(=int)	16	-32768～32767
short int	短整数	16	-32768～32767
unsigned short int	无符号短整数	16	0～65535
signed short int	有符号短整数(=int)	16	-32768～32767
long	长整数(=long int)	32	-2147483648～2147483647

（续）

数据类型	类型描述	位数	取值范围
unsigned long int	无符号长整数	32	0～4294967295
signed long int	有符号长整数(=long int)	32	-2147483648～2147483647
float	浮点数	32	-3.4E+38～3.4E+38
double	双浮点数	48	-1.7E+308～1.7E+308
long double	长双浮点数	80	-3.4E+4932～3.4E+4932

当用 short、long、signed、unsigned 来修饰 int 时，关键字 int 可以省略，例如无符号整型 unsigned int 可简写为 unsigned。

一个字符型(char)的对象占用计算机的一个字节，字符型对象存储的是该字符的机器字符代码(PC 中为 ASCII 码)，在 C 语言中认为字符是一个“小”整数。一个整型(int)对象典型情况下占用一个机器字(word)，C 语言提供 short 和 long 这两种整型数据类型，分别为短整型和长整型，所以 char、short、int 和 long 也统称为整型。这些类型可以是有符号的(signed)，也可以是无符号的(unsigned)。长整数和双精度数有助于存放更大的数，但是计算机必须为这些类型花费更多的存储空间和计算时间。

由于字符的存储形式是该字符的 ASCII 码值，所以在表 1-1 中字符型的存储值范围用数据值表示。从这种意义上讲，在 C 语言中整型和字符型可以互用。如果将一个整数存放在一个字符型变量中，就是在这个字符型变量中存放了该整数作为 ASCII 码值所对应的字符，在处理中将以该 ASCII 码值所对应的字符作为处理对象；反之，若在一个整型变量中存放一个字符，则在这个变量中存放了该字符对应的 ASCII 码值，以后的处理将对这个整数值进行。所以下面的表示都是有意义的。

```
int i=1;                int j='1';
char ch1='1';           char ch2=65;
```

i 中存放的是整数 1，j 中存放的是字符‘1’的 ASCII 码值 49(十进制)。ch1 中存放的是字符'1'的 ASCII 码值 49(十进制)，而字符型变量 ch2 中存放的是整数 65，它实际为字符'A'的 ASCII 码值。在字符型变量中只能存放一个字符，不能将由多个字符组成的字符串放到一个字符型变量中。C 语言中没有字符串类型，这是 C 语言和许多其它程序设计语言不同的地方。关于字符串将在后面讨论。

1.2.3　变量

其值可以改变的量称为变量。变量的作用是存储程序中需处理的数据。在计算机内存中，每个命名的变量有一个存储区域，区域内存放的是变量的值。可以在程序的任何位置定义变量，但必须先定义后使用(初学者应注意！)。变量有三个特性：

1) 每个变量有一个名字。

2) 每个变量属于一种类型，类型决定了对该变量所能执行的运算。

3) 每个变量只能存放其类型允许的值。

变量名是变量的标识，应当是一个合法的 C 语言标识符。

【例 1-2】 编制一个将英寸(inch)转换为厘米(cm)的程序。

```
/*文件名：INCHTOCM.C；该程序将英寸(inch)转换为厘米(cm) */
#include <stdio.h>
```

```
main( )
{ int inch;                    /*以下三行定义了三个变量*/
  float cm;
  char equal;
  equal='=';                   /*用'='作为字符变量 equal 的值*/
  scanf( " %d " ,&inch);       /*输入英寸值*/
  cm=inch*2.54;                /*计算为厘米值*/
  printf( " inches %c %d\n " , equal, inch);
  printf( " cm %c %f\n " , equal, cm);
 }
```

在上述程序中定义了三个变量：inch、cm 和 equal，变量的定义是通知编译系统，要在计算机内存中保留存放三个值的空间，三个变量的类型由变量名前的 int、float 和 char 分别说明为整型、浮点型和字符型，即这三个变量将分别能存储一个整数、一个浮点数和一个字符。类型说明是 C 语言的关键字，变量名是用户标识所定义变量的标识符。

在 C 语言中允许在定义一个变量的同时为其指定一个初值，这称为变量的初始化。对在函数内定义的变量，未经初始化则没有一个确定的初值，例如在函数中的 int i; 语句执行后 i 的值是不确定的，这是因为 C 语言在为函数中的变量分配存储位置时并不同时将该存储区清空。初始化的方法是：用符号“=”为所定义的变量给一个初值，例如：

```
void main ( )
{ char first=' G ', middle=' M ', last=' S ';
   …
}
```

C 语言希望变量的类型与其存放的数据相匹配，例如不能将浮点数值赋给一个字符型变量。

1.2.4 常量

常量是指在程序中直接给出的数值及字符。在 C 中除了常量外，还有一种在程序中不能变化的常变量，常量和常变量是两个不同的概念。在此讨论常量，常变量将放在后面讨论。

1. 数值常量

C 的数值常量包括整型常量和浮点常量两种。

整型常量是指不包含小数点的整数，浮点常量是指包含小数点的数。整型常量如以 0 开始，表示一个八进制的数；以 0X 开始，表示一个十六进制的数。例如：

12	表示十进制的 12	
012	表示八进制的 12，	即十进制的 10
0X12	表示十六进制的 12，	即十进制的 18
−0X12	表示十六进制的−12，	即十进制的−18

当程序中使用一个整型常量时，C 语言将这个整数解释为能存储这个整数的最小类型(最小为整型)，例如整数 13 作为有符号的整型数而不是作为长整数对待。但如果将“L”加在整数的后面，C 语言将这个数作为长整型常量，例如 13L 是长整型常量。将“U”加在整型常量的后面，C 语言将这个常量作为无符号常量，所以 13U 表示无符号常量，而 13UL

表示无符号长整型常量。

浮点常量只支持十进制。在浮点常量后面也可以加上字母“F”及字母“L”，明确要求将该常量用一般的浮点形式存放，或是说明为一个双精度浮点常量。例如 5.67L 表示一个双精度浮点常量。

浮点常量可以用科学表示法表示，将一个数表示为一个尾数和用指数部分表示的 10 的幂的积，例如 3El0、0.9E-34 等。表 1-1 中用科学表示法表示浮点数的范围，其中长双浮点数存储的范围为$-3.4\times10^{4932}\sim3.4\times10^{4932}$，就其绝对值而言，双精度浮点数存储的绝对值最大为 3.4×10^{4932}，绝对值最小为 3.4×10^{-4932}。

2. 字符常量

字符常量表示一个 ASCII 码字符，需要用单引号括起来，例如‘1’表示了字符常量 1，而不是数值量。然而有些字符是无法用键盘上的字符来表达的，例如标准 ASCII 码的码值在 31 以前的字符、码值为 l27 的字符和扩展 ASCII 码的字符。C 语言提供了特殊的转义字符来表示这些字符，转义字符由反斜杠和后面的字符(或数字)组成，字符(或数字)经转义代表了新的含义。例如已经在前面例子中使用过的转义字符\n 表示新行。转义字符见表 1-2。

表 1-2　转义字符

转义字符	含　义
\a	报警(响铃)
\b	退格
\f	走纸(用于打印机)
\n	新行(回车和换行)
\r	回车
\t	水平制表符
\v	垂直制表符
\\	反斜线
\?	问号
\'	单引号
\"	双引号
\ooo	字符的八进制表示(o 为八进制数字)
\xhh	字符的十六进制表示(h 为十六进制数字)
\0	空字符

采用\ooo 或\xhh 的方式可以用一个八进制数或十六进制数来表示一个特殊字符。例如为了在程序中引用希腊字母，要使用前缀\x,然后是该字符的十六进制的 ASCII 码值。下面是一个在字符变量 delta 中存储希腊字母 δ (δ 的 ASCII 码值为 235，等于十六进制的 EB)的例子：

delta＝‘\xEB’

尽管在单引号中有四个字符，但这四个字符只代表了一个 ASCII 码字符，所以变量 delta 中也只存储了一个字符。

表 1-2 中的转义字符必须小写，大写只表示其自身，所以将 δ 写成‘\XEB’是错误的。如果在反斜杠“\”后面不是表 1-2 中的转义字符，则忽略反斜杠符号，而作为一个一般的

符号对待。

3. 字符串常量

字符串是若干字符的有序集合，用双引号括起来，例如人名和地名(例如"Beijing ")都是字符串。

C 语言的基本类型中只有字符类型而没有字符串类型。字符串的存储和处理是通过字符数组来解决的。字符数组将在后面讨论。下面是一些字符串常量的例子：

" C Programming "， " 123 "， " Southeast University "， " X "

字符串常量以 ASCII 码的'\0'值结尾。一个字符串常量长度就是组成字符串所有字符的总个数，而不包括字符串的终结符在内。例如 " Hello! " 的长度是 6 而不是 7。

字符 ' X ' 和字符串 " X " 是不同的，字符 ' X ' 存储 1 个字节的 ASCII 码值，字符串 " X " 存储为 2 个字节的字符串，除 X 的 ASCII 码值外，还有一个 ASCII 码值'\0'。

1.2.5 const 常量(常变量)

const 是一个类型修饰符，在 C 语言中，一个变量定义时被 const 修饰后，该变量中放的值不能被程序所修改，这个变量也成了"一个不能变的变量"。这种变量可以称为常变量。在定义常变量时必须同时用初始化的方法给以初值，否则将再也没有办法将一个值放到常变量中去。C 语言推荐用大写字母作为常变量的变量名，以和一般的变量相区别。

【例 1-3】 编写求圆面积的程序。

```
/*文件名：CONSTEXA.C；说明 const 变量的使用*/
main( )
{
    float radius ;
    const float PI=3.14159 ;
    scanf( " 请输入半径:%f\n " ,&radius);
    printf( " 圆面积为:%f\n " ,radius*radius*PI);
 }
```

下面的程序段违反了对常变量的操作方法，所以是不正确的：

```
const int age ;
age=25;
const float salary=756.80;
salary=salary*1.1;
```

const 常量用在不希望由程序修改变量值的场合。例如数学中的π是一个很好的例子，当π的值放在一个变量中后，显然不希望程序对该值进行修改。

1.2.6 预处理指令

在前面已经提到，编译系统不是在编译源程序文件时立即取得其源代码的，而是由称为预处理器的预编译器首先查阅源代码，并对其中的预处理指令进行处理。

预编译器是编译系统的前端部分，预编译器对源程序中的预处理指令处理后，将经过处理的源程序文件给编译系统去做真正的编译工作。预编译器并不做实际的编译工作。

预处理指令并不是 C 语言的命令或函数。预处理指令由'#'开始，在行尾也没有分号。由于 C 语言是自由格式的语言，预处理指令并不一定从第一行开始，不过常规是由

第一行开始写预处理指令。预处理指令可以出现在程序的任何位置。

C 语言有若干预处理指令，本书介绍包容指令和宏定义指令，其它预处理器指令请参阅 C 语言手册。

1. 包容指令

#include 预处理指令称为包容指令，其作用是将一个磁盘文件合并到源程序中。换言之，#include 要求预编译器到磁盘上去找所说明的文件，并将其内容插到#include 所在的位置处。

include 有两种格式：#include <文件名> 和#include “文件名”

文件名所指的是称为头文件的文本文件，头文件的扩展名通常为“.h”。在头文件中一般是宏定义、外部变量声明及函数原型。第一种格式要求在编译系统预定的 include 目录中寻找所指定的头文件；第二种格式首先在程序源文件所在的目录中寻找所指定的头文件，如果没有找到，再到编译系统预定的 include 目录中寻找所指定的头文件。每一条包容指令只能包容一个文件，一个程序文件可能需多条包容指令来包容多个文件。在由包容指令所包容的文件中可以再有包容指令，形成嵌套包容。例如，下列几个是常用的头文件：

stdio.h：包括一般的输入输出函数。

math.h：包括数学运算函数。

string.h：包括字符串运算函数。

2. 宏定义指令

#define 预处理指令称为宏定义指令，其一般格式为：

#define 宏名 替换正文

#define 相当于字处理系统中的查找与替换命令，宏名是一个有效的 C 语言标识符，替换正文是任意的字符串。宏定义指令的作用是要求预编译器在源程序中寻找宏名，并将所有的这些宏名用替换正文来替换。字符串直接量中的宏名不被替换。

#define 的第一个作用是定义一个符号常量，从而使得程序使用的直接量符号化，并使其有一定的标识意义。

【例 1-4】 演示宏定义的使用。

```
/*文件名：DEFINEl.C：演示宏定义的使用*/
#define CONSTANT 125
main(  )
  {  int x;
     x=C0NSTANT-5;
     printf( " x 为:%d\n " ,x);
  }
```

在程序编译前，预编译器将程序中的所有符号常量 CONSTANT 替换成 125，经过预编译器处理后，程序在实际编译前已经不存在符号常量 CONSTANT，所以符号常量和 const 常量不同。

#define 的第二个作用是定义一个带参数的宏，带参数的宏在使用上很像一个函数，下面是一个求两个数乘积的宏：

#define MULTI(a,b) a*b

在程序中可以这样来引用这个宏：MULTI(3,7)。如果程序中有两个变量 x 和 y,还可以

这样来引用这个宏：MULTI(x,y)以及 MULTI(x+3,y+7)，此时需要将这个宏定义修改为 #define MULTI(a,b) ((a)*(b))，以得到正确的值。

1.2.7 输入和输出

C 语言中输入及输出分别由标准库函数 scanf( )和 printf( )完成。

1. C 语言输出函数 printf( )

printf( )函数的一般格式为：printf(格式控制串，输出项序列)

在格式控制串中允许包含两类成分：格式转换说明符、普通字符和转义字符，如表 1-3 所示。格式转换说明符控制输出项序列中各个输出项的输出格式，普通字符按原样输出，转义字符按前述的转义格式输出。例如：

printf("半径＝%d 厘米，圆面积＝%6.2f 平方厘米\n"，radio，radio*radio*3.14159);

其中 radio 是的一个整型变量，在格式控制串中%d 和%6.2f 是格式转换说明符，其余都为普通字符和转义字符。输出项序列包含变量 radio 和表达式 radio*radio*3.14159，多个输出项之间用逗号分隔。格式转换说明符按排列位置与输出项对应，分别将 radio 的值和表达式计算的结果值转换为整数和浮点数输出，浮点数输出为 6 位(其中小数点计为 1 位，小数部分输出为 2 位)。设 radio 的当前值为 2，该语句执行后将输出下列的结果并换行：

半径＝2 厘米，圆面积＝12.57 平方厘米

表 1-3 格式转换说明符

格 式 符	说　　明
%d	以带符号的 10 进制形式输出整数(以实际长度输出，正号不输出)
%o	以带符号的 8 进制形式输出整数(不输出前导 0)
%x	以带符号的 16 进制形式输出整数(不输出前导 0x)
%u	以无符号的 10 进制形式输出整数
%c	以字符形式输出一个字符
%s	输出字符串
%f	以小数形式输出单、双精度数，隐含输出 6 位小数
%e	以指数形式输出单、双精度数，数字部分小数位数为 6 位
%g	选用%f 或%e 输出宽度较短的一种格式，不输出无意义的 0
l	用长整数形式输出，可加在 d、o、x、u 的前面
m(正整数)	指定输出数据所占宽度(含小数点)
.n(正整数)	对实数表示输出 n 位小数，对字符串表示截取的字符个数

【例 1-5】 数据输出说明。

```
#include "stdio.h"
main( )
{ unsigned int a=65535;
  int b=-2;
  float f=123.456;
  printf("a=%d,%o,%x,%u\n",a,a,a,a); /*分别输出 a 的十进制、八进制、十六进制、无符号十进制数*/
  printf("b=%d,%o,%x,%u\n",b,b,b,b);
  printf("f=%f,%10f,%10.2f,%.2f\n",f,f,f,f)
```

```
    }
```

则输出结果为：

```
a= -1, 177777, ffff, 65535
b= -2, 177776, fffe, 65534
f= 123.456001, 123.456001,      123.46, 123.46
```

2. C 语言输入函数 scanf( )

scanf( )函数的一般格式为：scanf(格式控制串，地址项序列)

格式控制串和 printf( )函数中的格式控制串类似，所不同的是在由其中的格式转换说明符对输入格式进行的控制中，普通字符是指要求输入的普通字符，而很少用到转义字符。地址项序列由若干个等待输入变量的地址所组成，地址项间用逗号分隔。这里特别强调接受输入的输入项必须是一个地址量，例如要通过输入获得变量 radio 的值，该输入项必须写成&radio，“&”是 C 语言中取地址运算符，它表明将输入的数据放入运算符后面的变量所在的内存单元中。多个输入地址项之间用逗号分隔，格式转换说明符按排列位置与输入项对应。格式转换说明符和表 1-3 中类似，只是不包括不适合做输入的 u、g。

【例 1-6】输入三角形的三个边长，计算三角形的周长和面积(设输入的三个边长能构成一个三角形)。

```
/* 文件名：INOUT.C  */
# include<stdio.h>
main( )
{ float a，b，c，s，area ；
  printf (" Please input three line：");
  scanf （" %f%f%f"， &a,&b,&c）；
  s＝1.0/2.0*(a+b+c);
  area＝aqrt(s*(s-a)*(s-b)*(s-c));
  printf(" a=%7.2f,b=%7.2f,c=%7.2f,s=%7.2f" ,a,b,c,s);
  printf(" ,area=%7.2f" ,area);
}
```

3. C 语言的字符输入函数 getchar( )和字符输出函数 putchar( )

这两个函数的作用分别是从键盘得到一个字符及将一个字符送到屏幕上，两个函数的原型分别是：

char getchar( )和 putchar(char ch)

即 getchar( )函数得到一个字符并将该字符返回，而 putchar(char ch)是将在 ch 中的字符送到屏幕上。

【例 1-7】

```
#include "stdio.h"
    main()
    { char c;
      c=getchar();
      putchar(c)
    }
```

运行时，若输入字符 a，则输出结果为 a。

1.2.8 注释和缩进

C 语言是相当精致和灵巧的语言，C 语言只有 32 个关键字(C++在 C 语言的基础上增加到 48 个关键字)，同时 C 语言又是拥有最多运算符的语言之一。用 C 语言可以设计出相当精巧及高效的代码。但由于同样的原因，也导致 C 语言的程序比较难读。C 语言的灵活性容易导致一种追求程序设计技巧的倾向。程序设计的目的不是去创造一件“工艺品”，程序的可读性与可维护性比减少程序的代码长度更重要，朴素、清晰是程序设计中应当时刻牢记的准则。

在提高程序的可读性方面，程序的书写风格会起到很大的作用。程序的书写风格包括程序的书写格式，程序中的注释和说明，变量名、函数名及文件名等用户的命名习惯等等。

缩进是指程序在书写时不要每一个程序行都由第一列开始，而且在适当的地方要加进一些空格，要以醒目的方式表示出括号之间的配对关系等。

注释的目的是为提高程序的可读性，便于程序的理解和算法的交流。编译系统并不去理会注释的含义，只简单地将每个注释作为白字符处理。C 语言的注释以“/*”开始，以“*/”结束，“/*”和“*/”必须成对出现，可以同时跨越多行。C++的注释以双斜线(//)标志开始，该注释一直延续到这一行的行尾。在 C++中也保留了 C 语言的注释写法。

必要的注释是指下列内容：

1) 对于每个源程序的注释，一般应说明它的内容，引用了哪些资料，为维护本文件的总的提示等。

2) 对每个较大的函数，注释应陈述其目的、所用的算法、以及对有关环境所做的假设。

3) 在程序代码不清晰以及不可移植之处应有少量的注释。

4) 极少量的其它注释。

1.3 运算符和表达式

和其它程序设计语言一样，C 语言中记述运算的符号称为运算符，运算的对象称为操作数。对一个操作数进行运算的运算符称为单目运算符，对两个操作数进行运算的运算符称为双目运算符，三目运算符对三个操作数进行运算。

C 语言是拥有运算符最丰富的语言之一。运算符的含义取决于操作数的类型。例如当操作数是浮点型(float)时，a+b 中的“+”是浮点加；如果是整型(int)时，就是整型加。有些运算符的含义还取决于操作数的个数，例如&作为单目运算符时是取地址运算符，作为双目运算符是字位与运算符。

由运算符和圆括号将运算对象连接起来的有意义的式子称为表达式。常量、变量以及函数调用本身是最简单的表达式。每个符合 C 语言语法规则的表达式的计算结果都将得到一个确定的值。一般而言，表达式结果的类型取决于操作数的类型，不同类型进行混合运算时，根据系统确定的类型转换规则经转换后求值。当由多个运算符组成复合表达式时，运算符的求值次序根据运算符的优先级和结合规则确定。

1.3.1　算术运算符

1. 基本的算术运算符

C 语言中基本的算术运算符有加、减、乘、除及求模五种，分别由＋、－、*、/及%来表示。

C 语言中算术运算和数学中算术运算的概念及运算方法是一致的，要注意的是：除法运算符“/”包括了浮点型除和整型除两种运算，当除数和被除数都是整型数时，结果只保留整数部分而自动舍弃小数部分；除数和被除数只要有一个是浮点数，则进行浮点数相除。求模运算符又称为求余运算符，是求两个数相除的余数的运算，要求两个操作数都是整型数，如 6%2 的结果是 0，5%2 的结果是 1。

2. 增 1、减 1 运算符

增 1 运算符＋＋和减 1 运算符－－是两个单目运算符，它们的操作数类型只能是整型或字符型。增 1 运算符是将变量的值增加 1，减 1 运算符是将变量的值减 1。

需要注意的是增 1、减 1 运算符只适用于变量而不能用于常量或表达式。如：--5 或(a+b)++等都是不合法的。

它们可以位于运算对象的前面，如++a、－－b，这时要先对变量进行加 1 或减 1 操作，再使用变量的值。另外它们也可以位于变量的后面，如 a－－、b++，那么变量的值将在进行加 1 或减 1 操作之前被使用。因此在使用增 1 和减 1 运算符时必须注意其位置。

【例 1-8】 增 1 和减 1 运算。

```
x=2;                    x=2;
y=x++;                  y=++x;
结果 y=2,x=3            结果 y=3,x=3
```

合理地使用增 1、减 1 运算符可以使表达式简洁、明快，对流程控制语句及指针设置等的使用将带来很大的方便。此外，增 1、减 1 运算要比 a=a+1、a=a-1 的计算快。

3. 算术运算符的优先级

在一个表达式中如果有多个运算符，则计算是有先后次序的，这种计算的先后次序称为相应运算符的优先级，先进行优先级高的运算，再进行优先级低的运算。算术运算符的优先级是*、/、%运算符高，而＋、－运算符低。由于圆括号的优先级最高，使用左、右圆括号可以改变运算的处理顺序，圆括号里的运算总是最先进行。一般来讲，高级语言中的算术表达式的计算遵循着人们的习惯。

4. 算术类型转换

在 C 语言中，将字符型(char 类型)及各种长度的整型和实型统称为算术类型。进行算术运算(包括下面介绍的逻辑运算和关系运算)时，往往会遇到混合数据类型的运算问题，例如一个整数和一个浮点数相加就是一个混合数据类型的运算。C 语言通过算术类型转换的方法，自动创建一个临时变量，将值域“较小”类型的数据临时转变为值域“较大”类型的数据，以使双目运算符两个操作数的类型一致，运算结果的类型和“较大”操作数的类型相同，这种方法可以得出最可能的精确结果。在表达式计算完成后，临时变量将不复存在。

数据类型转换的另一种方法称为“强制类型转换”。 强制类型转换是指由程序员在程序中通过指定类型来超越 C 语言的缺省转换，临时将一个表达式的结果类型改变为一