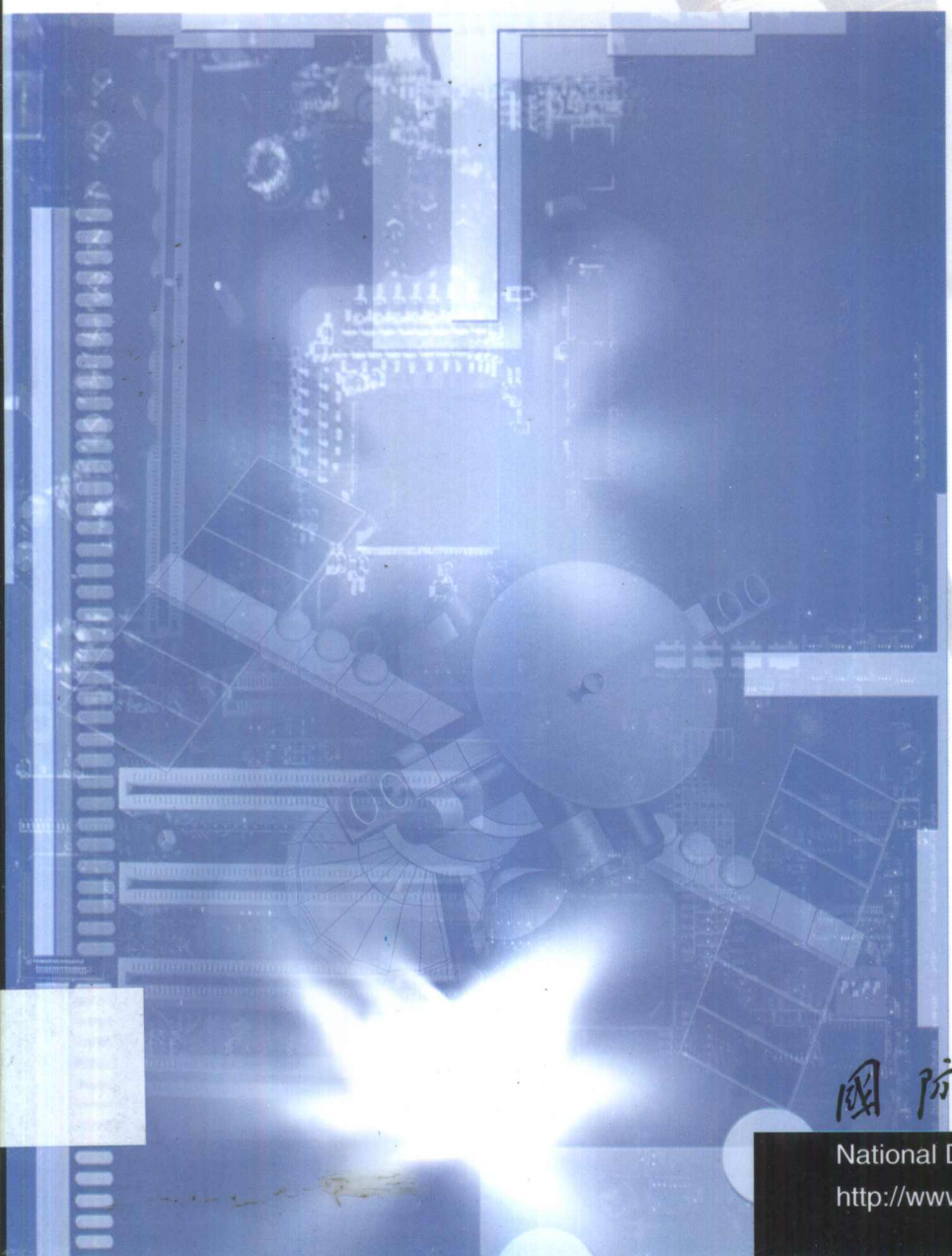
计算机应用基础

夏克俭 王成耀 李恪 编著



国防工业出版社
National Defence Industry Press
http://www.ndip.com.cn

计算机应用基础

夏克俭　王成耀　李　恪　编著

国防工业出版社

·北京·

图书在版编目(CIP)数据

计算机应用基础/夏克俭等编著 .—北京:国防工业出版社,2002.7
ISBN 7-118-02895-9

Ⅰ.计... Ⅱ.夏... Ⅲ.电子计算机-基本知识
Ⅳ.TP3

中国版本图书馆 CIP 数据核字(2002)第 047890 号

国防工业出版社出版发行
(北京市海淀区紫竹院南路 23 号)
(邮政编码 100044)
新艺印刷厂印刷
新华书店经售
*
开本 850×1168 1/16 印张 21½ 492 千字
2002 年 7 月第 1 版 2002 年 7 月北京第 1 次印刷
印数:1-3000 册 定价:29.00 元

前　言

目前，计算机在我国正在得到广泛而深入的应用，广大计算机工作者在分析和开发计算机系统软件和应用软件的过程中，迫切需要提高程序设计能力，掌握各种数据结构（或数学模型）的描述方法和处理算法，熟悉数据库系统的工作原理及设计方法，以便编制出实用的、高效率的计算机软件。因此，我们编写了《计算机应用基础》一书，可作为计算机相关学科的学生和计算机技术人员的专业基础书阅读。

本书共分为3篇。第1篇是“C语言程序设计基础”，共有5章：第1章介绍C语言程序的基本结构和C语言程序的框架；第2章介绍C语言中函数的定义与调用和建立结构化程序设计的概念；第3章介绍指针与数组；第4章介绍结构、联合及相关变量的定义与应用。第2篇是“数据结构”，共有9章：第1章介绍数据结构的基本概念、算法的定义及分析；第2章讨论线性表的定义及运算、线性表的顺序及链式存储结构、相关运算的算法实现以及线性表应用举例；第3章讨论栈和队列的定义及运算、栈和队列的顺序及链式存储结构、相关运算的算法实现以及栈和队列应用举例；第4章讨论字符串的定义及运算、字符串的顺序及链式存储结构及相关运算的算法实现；第5章讨论多维数组的表示、数组的存储映像、矩阵的压缩存储、广义表的定义和存储结构；第6章讨论树的定义及运算、树和二叉树的性质、树的存储结构、遍历算法、树和森林以及二叉树应用举例等内容；第7章讨论图的定义及运算、图的存储结构、遍历算法、最小生成树等内容；第8章讨论顺序表、树表和Hash表的查找以及相应的算法；第9章讨论插入、交换、选择、归并排序以及相应的算法。第3篇是“数据库系统概论”，共有4章：第1章介绍数据库的数据管理方式及核心软件DBMS；第2章介绍关系型数据库系统的数学基础和标准语言SQL；第3章是关系数据库理论，讨论关系模式的评价标准；第4章讨论关系型数据库的设计方法。

本书是我们在多年的教学基础上编写而成。由于作者水平有限，书中不足之处及错误在所难免，恳请读者批评指正。

目　录

第 1 篇　C 语言程序设计基础

第 2 篇 数据结构

第 3 篇　数据库系统概论

第 1 篇　C 语言程序设计基础

C 语言最初是由 D. M. Ritchie 为编写 UNIX 操作系统而设计的，其主要特点是表达式简练、运算符丰富、使用灵活，并提供了某些接近于汇编语言的功能（如二进制位操作等）。C 语言是一种通用程序设计语言，由于它非常适合于编写操作系统与编译器等系统软件，因而又称为系统程序设计语言。

本篇简要介绍 C 语言程序设计的基本知识，以便使读者能顺利阅读书中用 C 语言描述的算法与数据结构。其中，第 1 章介绍 C 语言的基本数据类型、语句以及简单 C 程序的基本结构；第 2 章介绍函数以及标识符的作用域；第 3 章介绍指针、数组以及二者之间的关系；第 4 章介绍结构、联合与链表的实现。

第 1 章　C 语言程序的基本结构

本章主要介绍 C 语言的基本数据类型、运算符、表达式、语句、编译预处理以及简单 C 程序的基本结构，以便使读者了解 C 语言的基本概念、C 程序的基本形式与语法特点，并能进行简单的 C 程序设计。

1.1　基本数据类型

数据类型规定了数据的取值范围与可参加的基本运算。C 语言的基本数据类型包括整型、实型与字符型等。

1.1.1　整型

整型分为下列几类：

① 整型：类型名为 int。

② 短整型：类型名为 short int，通常简写为 short。

③ 长整型：类型名为 long int，通常简写为 long。

④ 无符号整型：类型名为 unsigned int，通常简写为 unsigned。

⑤ 无符号短整型：类型名为 unsigned short。

⑥ 无符号长整型：类型名为 unsigned long。

这些类型的区别在于所表示的整数范围不同，其数据长度依赖于特定机器或编译器，C 标准没有具体规定。其中，较常用的整型是 int、short 与 long。在大多数系统中，short

型占 2 个字节,long 型占 4 个字节,int 型在 16 位或 32 位系统中分别占 2 个或 4 个字节。因此,在 32 位系统中,long 与 int 通常完全相同。

无符号整型、无符号短整型与无符号长整型只能表示无符号数,不能表示负数。例如,若 short 型占 2 个字节,则无符号短整型表示的整数范围为 0 ~ 65535。

1.1.2 实型

实型又称为浮点型,主要分为下列两类:

① 单精度型:类型名为 float。

② 双精度型:类型名为 double。

float 与 double 的区别在于所表示的实数范围不同,通常 float 型占 4 个字节,double 型占 8 个字节。相比之下,double 类型较为常用。

1.1.3 字符型

字符型数据的类型名为 char。char 类型用来表示单个字符,占 1 个字节,其内部表示一般是字符对应的 ASCII 码。因此,char 型数据实质上是一个整数,可作为整数参加数值运算。

1.1.4 空类型

空类型是 C 语言的一种特殊类型,类型名为 void,其主要用途是作为函数的返回类型,以明确表示相应的函数不返回任何值。

1.2 运算符与表达式

1.2.1 常量与变量

在程序运行过程中值不会发生变化的量称为常量,而值可以发生变化的量称为变量。常量与变量是最简单的表达式。

1. 标识符

C 语言规定标识符只能由字母、数字与下划线组成,且不能以数字开头。标识符主要用来表示变量名与函数名等,不能与 C 语言的关键字(如 int、char 等)相冲突。在设计程序时,选择的标识符名最好能反映其表示的具体含义,以提高程序的可读性。

C 标识符是大小写敏感的,如 COUNT、Count 与 count 是不同的标识符。习惯上,符号常量以及由 typedef 定义的类型采用大写名称,变量采用小写或大小写混合名称,其中,循环控制变量常选择较短的名称,如 i、j 等。为了避免命名冲突,程序员最好不要使用以下划线开头的标识符,因为系统标准库中的某些函数与变量采用了这种命名方式。

2. 常量

常量包括常数与符号常量两种形式。

1) 常数

(1) 整型常数

整型常数包括下列几种形式：

- 十进制整数：如 100、-765 等。
- 十六进制整数：以 0x 或 0X 开头，如 0x20(值为 32)、0xff(值为 255)等。
- 八进制整数：以 0 开头，如 012(值为 10)、027(值为 23)等。
- 长整数：以 l 或 L 结尾，如 123l、10246L 等。

(2) 实型常数

实型常数包括下列两种形式：

- 十进制小数形式：如 0.12、25.68 等。
- 指数形式：例如，1.5e2 或 1.5E2 表示 1.5×10^2。注意，字母 e 或 E 之前必须有数字，e 或 E 后面的指数部分必须为整数，如 e3 与 2.1E3.5 都不是合法的指数形式。

(3) 字符常量

字符常量是指以单引号括起来的单个字符，如字符'A'、'a'、',' 等。此外，还有一种特殊形式的字符常量，在引号内以反斜杠 \ 开头，主要用来表示不可打印字符。例如：

'\n'：表示换行符。

'\r'：表示回车符。

'\\'：表示反斜杠 \。

'\''：表示单引号。

'\0'：表示空字符(即 ASCII 码为 0 的字符)，等价于整数 0。

'\xhh'：表示 ASCII 码为十六进制数 0xhh 的字符，如'\x20'表示空格字符。

(4) 字符串常量

字符串常量是指以双引号括起来的由 0 个或多个字符组成的字符序列，如"The C Programming Language"、""(空字符串)等。C 语言字符串在内部表示时，以空字符'\0'作为结束标志。例如，字符串"Hello\n"在内存的存储形式如图 1.1 所示。

'H'	'e'	'l'	'l'	'o'	'\n'	'\0'

图 1.1 字符串"Hello\n"在内存的存储形式

要特别注意字符与字符串之间的区别。例如，字符'a'与字符串"a"是不同的，前者相当于一个整数，而后者则是由字符'a'与'\0'构成的字符序列。C 语言没有专门的字符串类型。

2) 符号常量

所谓符号常量，是指以符号名表示的常量。符号常量由 #define 定义，其格式如下：

```
#define    常量名    常量值
```

例如：

```
#define    PI        3.14
#define    MAXSIZE   100
```

定义了两个符号常量 PI 与 MAXSIZE，分别等价于常数 3.14 与 100。

为程序中多次出现的某些常量定义一个符号名，这不仅有助于提高程序的可读性，而且便于程序的修改。

3. 变量

变量以名标识并可按名访问,变量名是由程序员定义的标识符。每个变量都有唯一的数据类型,在内存中占据一定的存储单元,在相应存储单元中存放变量的值。因此,要正确理解变量的两个要素:地址与值。

① 地址:表示变量在内存单元的位置。变量的地址是系统自动分配的,程序员一般不需要关心变量的具体地址。

② 值:即相应地址单元的内容。变量的值正是程序要处理的。

C是一种强类型语言,C程序中用到的所有变量都必须首先定义其类型,即"先定义、后使用"。变量定义的基本形式如下:

类型说明符 变量名表;

其中,变量名表可以包含一个或多个变量名,多个变量之间以逗号分隔。例如:

```
int    a, b, c;
```

定义了3个int型变量a、b和c;也可按如下方式分别定义:

```
int    a;
int    b;
int    c;
```

此外,变量可以在定义时初始化。例如:

```
int    i=0, j;
```

定义了2个int型变量i和j,并设置i的初始值为0。

1.2.2 基本运算符与表达式

表达式可以是一个常量、变量或者由若干运算符与运算分量构成。其中,只涉及常量的表达式称为常量表达式。常量表达式可以在编译时求值,凡是使用常量的地方均可使用常量表达式。

1. 算术运算符

算术运算符包括+(加)、-(减)、*(乘)、/(除)、%(取模)、++(增1)、--(减1)。

说明:

- 两个整数相除的结果为整数(取其整数部分)。例如,7/4的结果为1。
- 运算符%只能用于整数。例如,7%4的结果为3,即7除以4的余数。
- 对于整数相除与取模运算,若运算分量为负数,则结果与具体系统有关,通常应避免使用。例如,-7/4的结果可能为-1或-2,-7%4的结果可能为-3或1。
- ++与--出现在变量前后的含义有一定差异。例如,++i是先使i增1,再返回增1后的值,而i++是先返回i的值,再使i增1。

2. 关系运算符与逻辑运算符

关系运算符包括<(小于)、<=(小于等于)、>(大于)、>=(大于等于)、==(等于)、!=(不等于)。逻辑运算符包括&&(与)、||(或)、!(非),参加逻辑运算的运算分量为整型(非0表示真,0表示假)。

关系表达式与逻辑表达式的结果为布尔值,以整数表示(真为1,假为0)。例如,表达式:

'c' && 2 || (8 < 4)

的结果为1。

3. 位运算符

位运算符按二进制位进行操作,包括 &(按位与)、|(按位或)、~(按位取反)、^(按位异或)、<<(左移)、>>(右移)。参加位运算的运算分量以及运算结果均为整型。

按位与运算符 & 常用于屏蔽某些位。例如,设 ch 为 char 型变量,则

ch = ch & 0x0f

用来将 ch 的高 4 位清 0,其余位不变。

按位或运算符|常用于将某些位置 1。例如,设 x 为 int 型变量,则

x = x | 0x0f

用来将 x 低 4 位的每一位置 1,其余位不变。

按位异或运算符^用来在两个运算分量对应位不同时置该位为 1,否则清 0。一元运算符~用来求整数的反码,即将运算分量各位的 0 转换为 1、1 转换为 0。

注意位运算符与逻辑运算符之间的区别。例如,设 x、y 均为 int 型变量,若 x 的值为 1,y 的值为 2,则表达式 x & y 的结果为 0,而表达式 x && y 的结果为 1。

移位运算符<<或>>分别用于将左边的运算分量左移或右移,移位次数由右边的运算分量指定。例如,表达式 x << 2 用来将 x 的值左移 2 位,右边空出的两位填 0,即相当于 x 乘以 4。表达式 x >> 2 用来将 x 的值右移 2 位,若 x 为无符号量,则左边空出的两位填 0;若 x 为带符号量,则有些系统采用算术右移(将左边空出的位以符号位填充),而另一些系统则采用逻辑右移(将左边空出的位填 0)。

4. 赋值表达式

赋值表达式的基本形式为:

变量 = 表达式

其中,= 为赋值运算符,用来将右边表达式的值赋予左边的变量。

此外,C 语言还引入了下列复合赋值运算符:

+=、-=、*=、/=、%=、&=、|=、^=、<<=、>>=

使用这些运算符的赋值表达式采用下列形式:

变量 op= **表达式**

等价于

变量 = 变量 op **表达式**

例如,a += b 等价于 a = a + b。

赋值表达式的类型就是其左边变量的类型,并以赋值后变量的值作为赋值表达式的值。赋值运算符左右两边的类型通常要一致。

5. 条件表达式

条件表达式的形式如下:

表达式 1? 表达式 2: 表达式 3

其中,? 为一个三元运算符。若表达式 1 非 0(真),则条件表达式取表达式 2 的值,否则,取表达式 3 的值。

6. 逗号表达式

逗号表达式的形式为：

表达式 1，表达式 2，……，表达式 n

按顺序求出表达式 1、表达式 2、……、表达式 n 的值，以表达式 n 的值作为逗号表达式的值。例如，设 i、j 均为 int 型变量，表达式

$$i = (j = 3, ++j)$$

的值为 4。

7. sizeof 运算符

格式：sizeof **(类型名或变量名)**

功能：返回指定类型或变量的数据长度(字节数)。

1.2.3 类型转换

类型转换分为隐式转换与显式转换两种方式。显式转换常称为强制类型转换。

1. 隐式转换

在表达式中，实型、整型与字符型数据可以混合运算，但当参加运算的几个运算分量具有不同类型时，系统在计算前要自动进行类型转换。

对于二元算术运算符，若两个运算分量的类型不同，则系统将低级别的类型向高级别的类型转换。例如，若两个运算分量的类型分别为 double 与 int，则自动将 int 型分量转换为 double 类型；若两个运算分量的类型分别为 long 与 char，则自动将 char 型分量转换为 long 类型，等等。

对于赋值运算符，若两侧的类型不同，则系统自动对右边表达式的值进行截取或扩展。例如，若将实数赋给整型变量，则截取整数部分；若将 long 型的值赋给 int 型变量，则截取其低位部分。因此，将高级别类型的运算分量转换为较低级别类型时，可能会丢失信息，在使用时一定要注意。

2. 强制类型转换

形式：**(类型名)表达式**

功能：将表达式转换为相应类型。例如：

```
(float) i
(int) (x + y)
```

实际上，强制类型转换使用了一个一元运算符，形式为(类型名)。在强制类型转换后，原来变量的类型并未发生变化。

1.2.4 表达式求值顺序

在 C 语言中，表达式的求值顺序由优先级与结合性确定。优先级决定了不同运算符求值的先后次序，优先级高者先求值。当一个运算分量两侧的运算符具有相同优先级时，求值顺序由结合性来确定，分为左结合(从左到右求值)与右结合(从右到左求值)两种方式。

运算符优先级与结合性见表 1.1。表中同一行的各个运算符具有相同优先级，第一行中的运算符优先级最高，越往下优先级越低。表 1.1 中的某些运算符将在以后章节中加以介绍。