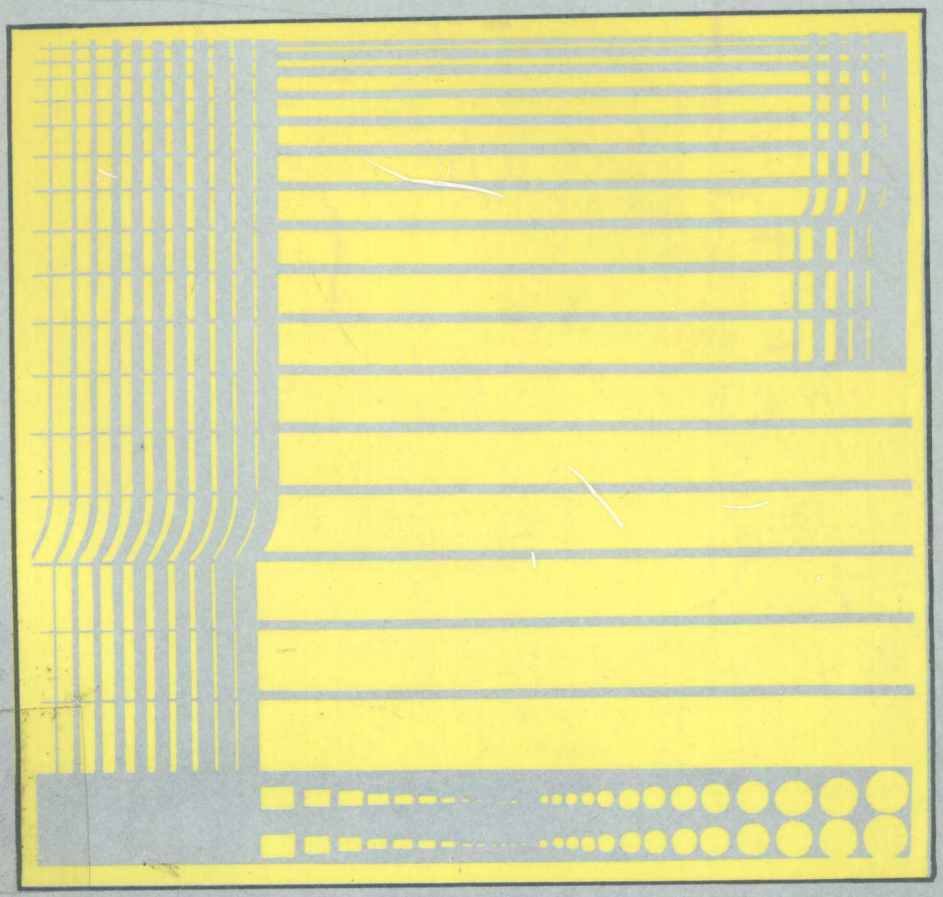
简明C语言教程

何向东　李敬安　赖康生　编



大连理工大学出版社

研究生教材

简明C语言教程

何向东
李敬安 编
赖康生

大连理工大学出版社

(辽)新登字 16 号

内 容 提 要

本书根据C语言的最新发展全面讲述了C语言的基本内容。其特点是融会了编者多年讲授C语言和用C语言编程的心得、经验，删繁就简，内容精练，例子典型，并附有习题解答。

本书可作为大学本科生及研究生学习C语言的教材，也适合作具有一定计算机语言基础的读者的自学教材。

简明C语言教程

Jianming C Yuyan Jiaocheng

何向东　李敬安　赖康生　编

大连理工大学出版社出版发行　　邮政编码：116024

大连海运学院印刷厂印刷

开本：787×1092　1/16　　印张：9 $\frac{1}{4}$　　字数：　211千字

1993年4月第1版　　1993年4月第1次印刷

印数：0001—2500册

责任编辑：路石　非文　　封面设计：羊戈

责任校对：寸土

ISBN 7-5611-0738-2/TP·41　　定价：4.80元

前　言

随着C语言的推广和普及，有关C语言的教材和专著越来越多。但一般教材的内容都太详尽、篇幅太多，常使初学者感到入门困难或望而生畏。本人根据多年讲授C语言的经验和用C语言编程的心得而编写出此书。其特点是紧紧把握住C语言的基本主线，删繁就简，叙述简练，例子典型，使篇幅大为缩减。多年教学经验证明，删去一些繁枝细节，不但不会影响学生掌握C语言的基本知识，反而能从一开始就掌握主动、步步深入，始终保持浓厚的兴趣。这种做法深受学生欢迎。

本书融会了编者和同事们用C语言编程的经验和心得，处处提醒初学者在使用C语言时容易犯的错误及要注意的事项，使初学者少走弯路。本教材还注重编程的训练，每章后面都附有大量习题。且遵循由易到难的原则，后面几章的习题已具有相当大的难度。对比较难的习题，书后还附有习题解答，供读者参考。

本书是以MicroSoft的版本(MSC)为根据讲授C语言的，但绝大部分是标准的C语言，可移植到其他C语言上。在用到MSC独有的、非标准的函数或宏替换时，书中会特别指出。因此即使不是用MSC的读者，只需把少数非标准的部分换成你使用的C语言中相应的函数或宏即可。

何向东

1992年10月

目　录

第一章　C语言简介

第一节　C语言的历史与现状

C语言是由贝尔实验室的D. M. Ritchie设计的，于1972年在小型机PDP-11上实现。当时C语言是为UNIX操作系统设计的系统语言。UNIX的研制者K. Thompson应用汇编语言和B语言于1970年发表了UNIX的初版。为了克服B语言的局限性，C语言应运而生。

B语言来源于BCPL语言，而BCPL是M. Ritchards于1967年发表的一种无类型系统程序语言。它的基本数据类型是机器字，指针和地址在该语言中用得很多。这跟结构程序的思想形成鲜明对照，结构程序是以使用强类型为特点。

C语言是在吸取这些语言的优点的基础上发展起来的一种成熟的通用语言。目前它已独立于UNIX系统，独立于PDP-11机，它适应的机种从8位机到巨型机。它与Fortran、Pascal语言一样，已经成了微、小、超小、大、超大和巨型机上共同使用的语言。

目前实用的C编译程序种类繁多。适用于IBM微机（及其兼容机）的C编译程序主要有MicroSoft C(MSC)、Turbo C(TC)和Lattice C(LC)，都可在PC-DOS或MS-DOS系统的支持下运行。C语言也在不断发展完善，例如MSC已推出7.1版本（目前最新版本），每推出一新版本，对C语言的功能就是一次发展和完善。

此外，各操作系统，如MS UNIX或XENIX系统，都有自己的C语言系统。其绝大多数函数或宏都是标准的，即通用的。但也可能有少数非标准的。在一种系统上能正常运行的程序不一定能顺利移植到别的系统上，对非标准之处需做相应的修改。

本书例题及习题解答中的所有程序，是根据MSC写的。绝大多数是标准的，全部都能在DOS操作系统支持下正常运行。少数用到MSC所独有的非标准函数或宏，书中都加以点明。

第二节　C语言的特点

C语言是一种小型语言。在程序设计中，小意味着简单易学，也意味着美。同时C语言又是公认的非常强有力的语言，它的强有力来源于精心选用的控制结构和数据类型。有的书称C语言是高级语言的低级版本以及低级语言的高级版本。这说明C语言同时具有高级语言与低级语言的优点。只要使用得当，C语言几乎具有无所不能的本领。这些年我们在使用C语言的过程中对这点深有体会，也充分享受到用C语言编程的愉快。

1. C语言的优点

(1)语言表达能力强，可以直接处理字符，数字，地址；可以完成通常要由硬件来实现

的算术运算及逻辑运算，几乎具有汇编语言的全部功能。

(2)具有数据类型的结构能力，具有很强的控制流结构。

(3)语言简洁，书写格式自由，可读性好。

(4)代码质量高，只比汇编低 10%～20%。

(5)可移植性好。

2. C 语言的不足

(1)早期的 C 语言没有浮点型运算，所有浮点运算都转化为双精度运算，运算速 度慢。MSC 6.0 以上版本已克服此缺点。

(2)运算符较多，不容易记忆。

(3)数据类型的转换较随便，容易出错。

第三节　简单 C 语言的程序

在深入学习 C 语言之前，先举一个简单而又比较完整的 C 程序，使读者了解 C 程序的构成与概貌。

例 1-1　简单的 C 程序。

```
//small.c  An  example  of  C  program
  main()                                          //主函数
  {                                               //开始
     int a,b;                                     //定义变量
     float c,sum;
     scanf("%d",&a);                              /*输入语句*/
     b=2;c=3.0;sum=a+c/b;                         /*赋值语句*/
     printf("Sum=%f",sum);                        /*输出语句*/
  }                                               /*结束*/
```

(1)凡是用一对注释符/*和*/括起来或有//的行都是注释。C 语言的注释可以写在任何需要注释的地方。注释不属于程序本身的内容，只是为读程序的人理解程序而加的。编译时对注释跳过不管。/*是注释的开始，*/是注释的结束。而//是注释开始，直到这行结束。左注释符/*与右注释符*/之间可以是任何文字、程序代码等，但不能再夹有注释。两种注释都可使用，本书都用//。

(2)main()是主函数，C 程序总是从主函数开始执行。因此一个可执行的 C 程序必须有且只能有一个主函数。

(3)一对大括号是函数的开始与结束。

(4)第 4、第 5 行为定义变量语句。经定义后，程序为这些变量开辟内存空间，这些变量就已存在，本函数可以调用这些变量。定义一个变量一般需要有存储类型说明和数据类型说明。这里由于都是内部自动变量，其存储类型说明 auto 都省略(有关存储类型将在第五章再详细讲)，只有数据类型说明。关键字 int 说明 a 和 b 是整型变量，float 说明 c 和 sum 是浮点型变量。

(5)第6行是输入语句，即从键盘给变量a输入数值。C语言的所有可执行语句以及定义变量的语句都必须以分号"；"结束。

(6)第7行有三句赋值语句。C程序允许多条语句串写在同一行。

(7)第8行是输出语句，即在屏幕上打出sum的值。

(8)标准输入输出语句scanf()和printf()是C的库函数。C编译器在编译C的源程序时，凡在语句中碰到后面带圆括号的标识符，都一律把它们当函数或宏对待，留到连接时才到库函数或别的目标模块中找它们的出处。

例1-2　带函数的最简单的C程序。

```
main()
{
    nothing();                    /*调用函数 nothing()*/
}
nothing()                         /*nothing 函数*/
{ }
```

主函数只有一句，即调用函数nothing()，而函数nothing()是一空函数。

第四节　C语言的编辑、编译、连接和执行

1. 编辑

C的源文件(源程序)可用edlin或wordstar或其他任何编辑程序来编写。文件名一定要用c做扩展名，如small.c，program.c等。由于C语言的书写格式较自由，因此用全屏幕操作的wordstar来编写C程序将显得特别方便。

2. 编译

源程序编写完后，需经C编译程序进行编译，生成机器代码，才能被计算机所认识。这是除BASIC以外，其他高级语言的共同特点。MSC编译程序有四遍扫描，第一遍为预处理，第二遍检查语法错误，第三遍产生代码，第四遍为代码优化。

为简化编译的手续，可用批处理文件进行。在MSC 6.0版本的软件中，提供建立编译批处理文件的详细说明。根据这说明可把编译时需要的所有文件装入到合适的路径，再建立一个批处理文件来调用这些文件，使整个编译工作自动进行。

我们所建立的编译批处理文件，取名c.bat，编译时只需在DOS系统下键入：

```
c  xxx <r>
```

即可。其中xxx为源文件名(不必带扩展名)，<r>表示回车。批处理文件自动完成编译所需的全部过程。如没错误，编译结束后，回到DOS系统；如有错误，则指出错误。需再用编辑程序修改源文件后再编译。

编译后生成的文件是目标文件，名字与源文件名相同，但扩展名是obj.

3. 连接

C也像其他高级语言一样，编译后的目标文件必须进行连接。例如程序用到的一些外

部过程(最通常情况是用到一些库函数)必须通过连接的手续将它们找到并加入到用户程序中,才能生成完整的可执行程序。C 的目标程序可用普通的 link 程序连接。为简化手续,我们同样把连接过程也建立批处理文件,取名 l. bat。连接时只需在 DOS 系统下键入:

```
l  xxx <r>
```

即可。这里 xxx 为目标文件名,可不写扩展名。如果连接没有错误,回到 DOS 系统。生成的文件为可执行文件,名字与目标文件名相同,但扩展名为 exe。

如果要同时连接几个目标文件,则在 l 后面把这些目标文件都列上,如

```
l xx1 xx2 xx3 .... <r>
```

最多可同时连接 9 个文件。生成的可执行文件,以第一个目标文件名命名。连接一般不会出什么错误,但当一些外部过程没找到时,连结结束后会指出这些错误。这种错误有时不是致命的,程序可以局部正确执行。有时是致命的错误,它使程序无法正确执行。此时必须找出原因,如果错误在源程序中,例如函数名写错,则需修改源程序,重新编译连接;如果是由于连接时少连某个目标文件(多个目标文件连接的情况),则只需重新按正确要求连接即可。

还要提及的是,MSC 6.0 有多种使用模式:小模式(S)、中模式(M)、大模式(L)、紧凑模式(C)及巨模式(H)等。可根据程序的规模选取其中的一种。小模式运用时,要求程序代码长度不能超过 64k 及数据也不能超过 64k;中模式运用时,要求程序代码长度可以超过 64k,但数据不能超过 64k;大模式运用时,程序代码长度和数据都可超过 64k。

当程序较短,数据量不大时可用小模式;相反的情况可用中模式或大模式。使用哪种模式,由连接时与哪个模式的库连接决定。例如想用小模式时,目标文件应与小模式的库相连;用大模式则与大模式的库相连。我们建立的连接批处理文件是连接大模式的。

4. 执行

源文件经编译连接生成可执行的 exe 文件后,在 DOS 系统下,键入可执行文件名(不必扩展名)即可。

```
xxx <r>
```

习 题

1-1 练习用 wordstar 或其他编辑程序编写例 1—1 的源程序。然后练习编译、连接、执行等过程。

1-2 在源程序中故意制造些错误,观察编译时给出的错误信息。

1-3 若源程序的文件名不加扩展名(.c),观察编译时给出的错误信息。

1-4 若将例 1-1 的源程序写成如下形式:

```
main(
{                              int  a
;b                            ;float  c,
sum;  scanf("%d",&a);          b=
```

```
2;  c=                3.0;  sum=a
+c/                       b;printf(
"sum = %f",               sum);}
```

你估计这个程序能顺利地通过编译、连接，并能正确运行吗？从中你能得出 什么结论？你认为这程序的可读性如何？能否提倡这种写法？

1-5 在修改程序时，如果有一段代码准备删去，但以后又可能还会改回原样。如果删掉了，当改回原样时又得补写回去，浪费时间。在这种情况下，可用注释符把这段代码注释掉。如果将来需要改回来，只需把注释符去掉即可。特别在对程序做试验性的修改，而这段代码又很长时，这样做特别省时省事，值得提倡。在程序修改定型后，再把不需要的代码真正删除掉，免得影响可读性。但上述做法要注意注释的有关规定。下面两个例子中，左边的程序是正确的，请分析哪些语句被删除了。右边的程序却有错误，请根据第三节有关注释的规定，分析错在那里？

```
main()
{
    int  a,b;
    oat  c,d,e,sum;
    scanf("%d",&a);
/*  scanf("%f",&d);
    e=d+a;
*/  b=2;c=3.0;sum=a+c/b;
    printf("Sum=%f",sum);
/*  printf(" d =%f",d);  */
}
```

```
main()
{
    int  a,b;
    float  c,d,e,sum;
    scanf("%d",&a)
/*  scanf("%f",&d);  /*canceled*/
    e=d+a;           /*canceled*/
*/  b=2;c=3.0;sum=a+c/b;
    printf("Sum=%f",sum);
/*  printf(" d =%f",d);*/
}
```

第二章　运算符与数据

第一节　标 识 符

标识符用来标记常量、变量、数据类型、函数名等。C语言允许用作标识符的字符有：

(a)所有的英文字母(大小写共52个)；

(b)所有数字字符(10个)；

(c)下线符_(1个)。

C语言的标识符可归为三大类。

1. 关键字

常用关键字如下：

auto	break	case	char
continue	default	do	double
enum	else	extern	float
for	goto	if	int
long	register	return	short
sizeof	static	struct	switch
typedef	union	unsigned	while

这些关键字在C语言中有特定用途，不准用这些关键字作变量名、函数名等。随着C语言的不断发展，关键字也在扩充。如MSC较新版本的的关键字还有far，near，frotran，pascal等，由于不常用，等用到时再介绍。

2. 特定字

特定字符共7个：

define	undef	include	ifdef
ifndef	endif	line	

编译预处理时将使用这些特定字，它们同关键字一样，不准用作变量名，函数名等。

3. 一般标识符

组成C语言的一般标识符(变量名，函数名等)，有下列规则：

(1)必须以小写英文字或下线符开头；

(2)后面跟随的必须是小写英文字、或数字、或下线符；

(3)长度限制，低版本以8个字符为限，MSC较高版本则扩展到31个字符。

例如，下列标识符是合法的一般标识符

a　bc　_file　x123　xyz　small_12　c_language　settextposition

下列标识符则是不合法的一般标识符

a/b　5n　aBc　x&y　I'right'　static

最后一个标识符 static 是关键字，因此不能作为一般标识符。

第二节　常　量

1. 整型常量

整型常量可有三种制式：

(1)十进制　　如　−30，　175　　等；

(2)八进制(以 0 开头)　　如　036，　0253　　等；

(3)十六进制(0x 开头)　　如　0x1e，　0xaf　　等。

2. 长整型常量

在数字后加 l 或 L 则表示该数为长整型数，例如 123l，56789L 等。

3. 浮点常量

相当于 Fortran 中的实数。由整数和小数组成，也可用指数形式表示，如 2.0，3.1416，1E5，−2e3，1.5e−5 等。

4. 自定义常量

C 语言可用替换方法来定义任一常量，如：

```
#define  PI  3.14159
#deine  NUMBER  100
```

一般习惯上把替换写在程序前头，而且用大写字母，以便与普通的变量相区别，但用小写也可以。经这样定义后，在程序中不直接用 3.14159 或 100 而用 PI 和 NUMBER 代替。这种写法比直接用常数直观和方便。例如，当需要将 100 改成 60 时，只须在替换处改动一下就可。如果直接用常数 100，则修改时要将程序从头至尾检查一遍，既费时又容易漏改。

自定义常量在 C 语言中用得很多。例如用 EOF 代表−1，表示文件的结束；NULL 代表 0，表示指针操作出错等。在头文件 stdio.h 中有这些常量的定义。

5. 字符常量

将键盘的有形字符用单引号括起来，则为字符常量，如'a'，'A'，'1'，'0'，'&'''等。这些字符常量是有数值的。如果计算机用的是 ASCII 编码制，字符常量的数值就等于该字符的 ASCII 码值，例如'a'＝97，'A'＝65，'1'＝49，'0'＝48(ASCII 码表见附录 A)。

所以字符常量是小于 127 的整型数。双字符\0 的 ASCII 值为 0，这是个很重要的字符常量，它常被用作字符串的结束标志。

6. 字符串常量

一串字符用双引号括起来则为字符串常量(简称串)，如"x"，"I'm a Chinese."等。在字符串的结尾，计算机自动加上双字符\0，表示该字符串的结束。因此字符串常量的存储单

元要比实际的字符串个数多一个。如"x"占两个存储单元，"I am a Chinese." 字符数为15个(有3个空格符)，存储单元为16个。

由于双字符\0的ASCII值为0，因此可检验字符串的字符值是否为零来确定字符串是否结束。

'x'与"x"的区别：

'x'是字符x的ASCII码值，是一整型数，而"x"是字符x本身。

第三节 变 量

C语言的变量类型很丰富，其分类如下：

1. 整型变量

C程序中用到的所有变量都必须事先定义。在定义时都要进行数据类型说明。对整型变量，是用关键字int说明，如：

int a; int x,y; 定义变量a，x，y为整型变量。

整型变量是用得最多的变量类型。任何一台计算机，能表示的数值大小都是有限的。对于16位机，整型变量的存储单元为一字长(2字节或16比特)，最左边的比特(最高位)作为符号位。所以整型变量的数值范围为-2^{15}～$2^{15}-1$，即 −32768～32767。

2. 无符号整型变量

用unsigned int 说明，其中int可省略。如：

unsigned a; 表示a是无符号整型变量。

对16位机，无符号整型的数值范围为 0～65535。

3. 长整型变量

用 long int 说明，其中 int 可以省略。如：

long a; 表示a是长整型变量。

对16位机，长整型的数值范围为$-2^{31}\sim 2^{31}-1$。

需要补充说明一点：整型变量中还有一类短整型(short int)。short可以用在需要关心存储容量的场合。因为C编译器有可能给short安排的存储单元比int小，但这取决于所用的机器。例如VAX机给int安排的内存为32比特，给short安排16比特。这种情况下int与short是有区别的。但对16位机，两者的存储单元都是16比特，这时int和short就没有区别了。

4. 字符型变量

字符型变量用关键字char说明，如：

```
char  c;    表示c为字符变量。
```

字符型变量的存储单元为一字节(8比特)最左边的比特作为符号位。所以字符变量的数值范围为$-128\sim 127$。

5. 无符号字符变量

用unsigned char 说明，如：

```
unsigned char c;      c为无符号字符变量，其数值范围为0～255。
```

6. 浮点型变量

C语言的浮点变量，与Fortran的实数相同，但不存在隐含说明，定义时必须用关键字float说明，如：

```
float a;     float i;     a, i即为浮点变量。
```

对16位机，浮点变量的存储单元为两个字长，最左边的比特作为符号位。

7. 双精度型变量

定义双精度变量时用关键字double说明，如：

```
double  a;          a为双精度变量。
```

双精度变量也是浮点变量，不过其存储单元比浮点变量大一倍，为4个字长。因此有的书用long float 说明双精度。

第四节　类型混合运算的原则

各类型变量如按其存储单元的多少及数值范围的大小来排序，则其顺序如下：

char＜unsigned char＜int＜unsigned＜long＜float＜double。

算术表达式，例如x+y，计算出一个具有一定类型的值。例如，如果x和y都是int型，那么x+y的值也具有int型。但是如果x和y的类型不同，则x+y是一个混合表达式。假定x是int，y是float，那么x自动转换成float，x+y的值是float型。注意：存储在内存里的x的类型并没有转换，转换的只是x的临时拷贝。这种转换称隐式转换。

如果把此时的x+y值赋给某个变量，例如a=x+y，则赋值的结果取决于a的类型。如果a是int，则x+y的值截去小数部分，只把整数部分赋给a；如果a是float，则x+y的值按原样赋给a。

因此，当不同类型的变量进行混合运算时，遵循下列原则：

(1)低类型自动向高类型进行隐式转换；

(2)运算结果保留高类型；

(3)赋值时，把结果隐式转换成赋值号左边变量的类型。

需要指出，上面总结的三条原则有点简单化。但对整型、浮点型、双精度型的混合运算(也是最经常碰到的情况)，这些原则是精确的。

例 2-1　混合类型的运算。

```
int a;          float sum;          double b;
sum = a + b;
```

当 a 和 b 相加时，整型 a 隐式转换为 double，a+b 的值为 double。把这结果赋给 sum 时，由于 sum 是 float，所以 a+b 的值隐式转换为 float 再赋给 sum。

例 2-2　强制类型转换

```
int a, b;          float c;
c=(float)a/b;
```

如果程序设计者本来的目的是要将 a 与 b 的商(包括尾数)赋给浮点量 c，但由于 a 和 b 都是整型，它们相除时不会自动转换为浮点型，而是整数相除。这样赋给 c 的值往往是错的。为此需将 a 或 b 强制转换为浮点型。这里的(float) 就是强制类型转换运算符，它把 a 强制转换成 float。由于 a 被强制成 float，a/b 成了混合类型表达式。那么根据混合类型运算的原则，b 将隐式转换成 float，a/b 的值也是 float 并原样赋给 c。

在本例中，如果 a=2，b=3 则不加强制转换时 c=0；加强制转换后，c=0.666667。

还有一类容易出现错误的运算，初学者必须警惕。如：

```
long a;          int c1,c2;
a = c1 + c2;
```

上式当 c1，c2 之和不大于整型数的数值范围(与所用机器有关)时，不会出现问题。但如 c1，c2 之和大于数值范围时，不要认为由于 a 是长整型数，因此这个和能被正确地赋给 a。事实上，由于 c1，c2 都是整型，运算结果也是整型。因此当结果超出整型数范围，得到的是这个和的补码。赋给 a 的数当然也是这个补码的。如果想将 c1，c2 之和在超出后仍能正确赋给 a，应将其中一个强制转换为长整型。

如：

```
a = (long)c1 + c2;
```

第五节　运 算 符

1. 单目算符

(1) *a　　——取内容，其中 a 为指针

(2) &a　　——取地址

(3). 及→　　——取结构成员

(4) sizeof()　　——计算结构的长度

(以上四个算符留待讲指针和结构时再介绍。)

(5)－a　　——取负，即取 a 补码

(6)！a　　——逻辑非，若 a 为零，则！a 为 1；若 a 为非零(不必是 1)，则！a 为零

(7)～a　　——取 a 的反码，例如 a 的二进制码为 1011，则～a 为 0100

(8)前＋＋　——增 1，例如 ＋＋x 等效于 x=x+1

(9)前－－　——减 1，例如 －－x 等效于 x=x－1

(10)后＋＋　——增 1，例如 x＋＋ 等效于 x=x+1

(11)后－－　——减 1，例如 x－－ 等效于 x=x－1

前＋＋(或前－－)与后＋＋(或后－－)的区别：

a=＋＋x；　x 先加 1 再赋给 a；而

a=x＋＋；　x 先将当前值赋给 a，再自身加 1。

(12)(类型)　——强制类型转换

2. 双目算符

(1)算术运算符

＋(加)　－(减)　*(乘)　/(除)　%(取模)

%是取两个整数相除的余数。例如 5%3=2，10%5=0，－5%3=－2，5%－3=2 等。

注意：其中%只能作用于整型数或整型变量。

(2)关系运算符

＞　＞=　＜　＜=　==　！=

注意：

(a)关系算符的两边必须是同一类型的量，否则有时会出错；

(b)如果表达式成立，则表达式为 1，如果不成立，则表达式为零；

例如表达式 a==b，a，b 必须是同一类型；如果 a 确实等于 b 则表达式为 1；如果 a 不等于 b 则表达式为零。

(3)字位运算符

&(按位与)　|(按位或)　^(按位加)　＞＞(右移)　＜＜(左移)

注意：字位算符只能作用于整型变量或整型数。

字位算符作用于以二进制数字串表示的整型表达式上。它们显然与所用的机器有关。对于 16 位机，整型数内存单元位 16 比特。设整型 a=11，b=10，则其二进制数为：

a	0000000000001011	
b	0000000000001010	那么
a&b	0000000000001010	a 和 b 按位与，原则：11 得 1，有 0 则 0
a|b	0000000000001011	a 和 b 按位或，原则：有 1 得 1，00 则 0
a^b	0000000000000001	a 和 b 按位加，原则：按位加，不进位
a＜＜2	0000000000101100	a 的数向左移 2 位，右边以 0 填补，等效于 a＊4
a＞＞2	0000000000000010	a 的数向右移 2 位，左边以 0 填补，等效于 a/4，但要注意有的机器(如 VAX)左边以符号位填补

字位运算符很有用处，例如可用 & 对一个变量或一个数的某些字位进行屏蔽。如：