C/C++与数据结构

金以文 编著

浙江大学出版社

前　言

本书由三部分组成：C/C++和数据结构。

第一章概述了C/C++的发展历史。

第二、三章介绍了C语言。C语言是一种面向结构化的程序设计语言，功能强且使用灵活，是80年代颇受欢迎的程序设计语言，可以说在80年代到达了它的辉煌顶点。无论在大型的著名UNIX操作系统或者是各种事务处理软件，甚至在作为一些单片机汇编语言的工具等方面，C语言都成功地被人们所使用。如果熟悉C语言的读者，对这一章只需快速浏览一下即可，而对C语言陌生的读者，则应结合参考书认真学习。

第四章至第八章讲述面向对象的程序设计语言C++。C++是C的派生，是C的超集，使一个面向结构化的程序设计语言转向面向对象的程序设计语言(Object-Oriented Programming Language，简称OOPL)，目的是使程序设计者能更好地理解和管理庞大的复杂的程序。C++提出了一种全新的程序设计思想，毫无疑问，C++将改变人们编写程序的方式，在今后的几年里，C++将会得到进一步的普及。

第九章至第十三章介绍数据结构。“数据结构”是计算机软件的一门基础课程。本课程的目的是介绍一些常用的数据结构，例如：线性表、链表、树和图等。阐明数据结构内在的逻辑关系，讨论它们在计算机中的存贮方式以及相应的各种运算。在介绍数据结构基本概念的同时，给出了用C/C++编写的各种数据结构的算法程序。

本书有两个特点：

第一，将C/C++和数据结构合并成一门课程，这样，学生就不必选修三门课程。这是适应时代的需求，提高教学效率，既省时又实用。

第二，内容安排紧凑，层次清楚，深入浅出，通俗易懂，内容丰富。本书含有大量实例，读者通过实例能很快地熟悉概念，学会应用，提高程序设计能力。

本书对各类大专院校学生和非计算机专业的研究生，从事CAD和CAM专业的工程技术人员是一本比较合适的教材和参考书。通过本书的学习，能对C/C++和数据结构有较全面的认识，为独立进行有关领域的软件设计打下良好的基础。讲授学时为70学时左右。

最后说明一点，本书主要内容是介绍“C++”和“用C++讲授数据结构”，其中有关C语言和数据结构基本概念的介绍均取自本人编写的《C语言与数据结构》一书，详细内容可以参见此书。

本书在编写过程中，得到了鲁世杰教授和赵乃良副教授的大力支持，他们在文字阐述、内容安排和实例选择等方面都提出了许多宝贵的意见，在此表示衷心地感谢。

由于水平有限，书中不妥之处在所难免，热诚希望读者指正。

编　者

1996年9月

目　录

第一章　C/C++概述

第二章　C语言基本概念(一)

第三章　C语言基本概念(二)

第四章　C++对C的基本扩充

第五章　C＋＋基本要素

第六章　继　承

第七章　多态性

第八章 C++的I/O类库

第九章 数据结构概论

第十章 线性表

第十一章 链 表

第十二章 树

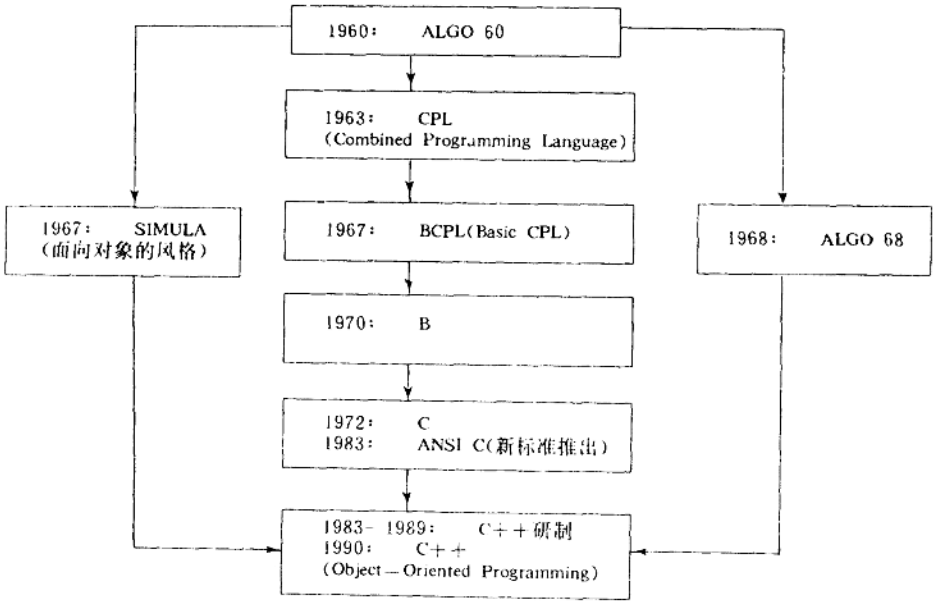
第十三章 图

第一章　C/C++概述

1.1　发展过程和特点

C语言是一种面向结构的程序设计语言，是一种功能很强而又比较简单的通用程序设计语言。它具有高级语言的许多特点，又具有比较低级的汇编语言的功能；既是成功的系统描述语言，又是一种实用的程序设计语言，这就是C语言具有的两重性。C是目前在国际上广泛流行且深受欢迎的语言之一。然而，随着对软件要求的提高，当软件达到一定规模时，它的局限性也越来越明显。一般来说，程序达到25000行至100000行规模时，它就变得相当复杂，很难合为一个整体，从而难以管理。为了解决这一问题，首先由美国新泽西贝尔实验室的Bjarne Stroustrup及其同事们于1980年对C进行了扩充，起初称这一个新语言为"带类的C语言"，在1983年改名为C++。随着C++的出现，它经历了两次大修订：一次在1985年，另一次在1989年。C++的有效性是它既保留了C的原始精华，又使一个面向结构化的程序设计语言变为面向对象的程序设计语言，使程序构造得更清晰，更易扩充和维护。

C和C++发展过程如下：



C语言具有以下特点：

(1)具有丰富的数据类型，较多的运算符，较少的关键字，简洁的表达方式，便于编写各种

应用程序；

(2)提供了良好的结构程序所需的基本流程控制语句。程序结构是以函数为基本的集合体，函数能独立编译和调试，一个程序就可以由小到大，由简单到复杂堆积而成，因此，适宜于大型程序的模块化设计；

(3)提供了指针，能做地址运算，能直接访问物理地址，使得它在直接操作硬件功能时，能实现同汇编语言相同的描述；

(4)通过预处理可以进行宏调用，可防止书写错误，便于修改常数；

(5)用C语言编程，程序员可以获得高效率的机器代码，其效率几乎接近于汇编语言的代码；

(6)C语言程序非常容易移植，基本上不作修改就能用于各种型号的机器。

C++除保留了C的以上优点外，还具有许多独到之处，主要特性如下：

(1)封装性

C++的封装性，是通过引入"类"(class)来实现的，类将一定的数据和对这些数据的操作代码封装在一起。这个特点减少了各模块之间的联系，在大型软件开发过程中，好处尤为明显。

(2)继承性

继承是由一个类获得另一个类的特性的过程。建立一个新的派生类，它从一个或多个先前定义的基类中继承数据和函数，并可以再定义自身的数据和函数，这样不但建立了类的层次，而且大大提高了代码的可重用性。

(3)多态性

C++的多态性，是指对目的相似而实现方法不同的操作可以用同一个名字。C++的多态性使C++与人的思维习惯更趋一致，用C++编制的程序也更容易阅读。

所以我们可以想见，今天的C++正如当年的C语言一样，会给软件设计带来一次新的巨大的进步。C++既具有C的一切优点又能进一步弥补C的不足，因而它比C优越，因此，C++已被许多专家公认为是近期内最有发展前途、并最为流行的现代程序设计语言。

1.2 C/C++的基本符号

基本符号有英文字母、阿拉伯数字和特殊符号。其中有：

(1)英文字母26个(包括大写和小写)；

(2)阿拉伯数字10个，即从0到9；

(3)特殊符号主要是指运算符，如：

初等量表达式操作符：( )，[ ]，.，->；

一元操作符：*，&，-，!，~，++，--，sizeof()；

二元操作符：*，/，%，+，-，>>，<<，<，>，<=，>=，==，!=，&，^，|，&&，||，

? :；

赋值组合操作符：=，+=，-=，*=，/=，%=，>>=，<<=，&=，^=，|=；

逗号操作符：，。

(4)标识符

用来表示函数、类型和变量的名称。由字母数字表示，必须以字母开头(连字符_也称为字母)，C语言标识符区分大小写，按一般习惯，变量名用小写字母，符号常数用大写字母。在C

中，通常只有前 6～8 个字符有效，ANSI C 标准要求前 31 个字符有效，而在 C++中对标识符长度没有特别限制。

(5)关键字标识符有：

- 描述数据类型定义的关键字：typedef；
- 描述变量存贮方式的关键字：auto，extern，static，register；
- 描述数据类型的关键字：char，short，int，long，unsigned，float，double，union，struct，enum；
- 描述语句的关键字：while，do，for，switch，break，continue，return，goto，if else，case；
- 其他：void，default 等。

C++新增以下关键字：

class，private，protected，public，this，new，delete，friend，operator，inline，virtual 等。

用户使用的变量名和函数名不能用以上的关键字标识符。

1.3 C/C++程序结构

用 C 语言编写的程序，称作 C 语言源程序，简称 C 程序。C 程序一般由一个或多个函数组成，这些函数可集中放在一个或分散在几个文件中。各文件都必须以.c 结尾，如 sum.c。

首先写一个简单的程序：

```
/* add of a and b */
    main()
    {       int  a,b,sum;
            a=123;
            b=456;
            sum=a+b;
            printf("sum is %d\n",sum);
    }
```

执行结果：　sum is 579

这个程序的功能是把 a 和 b 两个数相加，然后输出和的结果。对上述程序说明如下：

(1)/*与*/之间的内容是注释，注释行可以出现在程序的任何部分，注释界符/和*中间不能有空格，而*和注释之间要有空格。注释行不被编译，只起注解和帮助阅读的作用。

(2)main 是一特殊函数名，所有 C 语言程序都以 main 函数开始执行，其实质是 C 程序的首部(或主程序)，main()后面的圆括号()不能省去，参量放在括号内。该程序是一个无参量的函数。

(3)花括号{}是函数体的界限符，用以包括构成函数的全部语句。

(4)第 4 行是变量 a，b，sum 的说明语句，int 是 integer(整型)的缩写，说明 a，b，sum 是整型变量。

第 5、6 行是两个赋值语句，把整数 123 赋给变量 a，整数 456 赋给变量 b。

第 7 行是把 a 和 b 值相加，结果赋给变量 sum。

第 8 行输出变量 sum 的值。双引号"……"内所指出的字符串是控制格式，%符后面的字母为格式符，这里 d 表示用十进制格式输出，\n 是 C 中换行符。printf 是输出函数，是系统库提供的标准函数。

(5)每个语句必须用分号(;)结尾。一行中可以有若干个语句。有一点特别注意,花括号"}"后面不能有分号。

在一个完整的C程序中,可以由若干个函数组成,但一定要有一个函数main(),不管这个函数放在C程序中什么地方,C程序总是从它开始执行。函数main()可以调用其他函数,其他函数之间也可以有调用和被调用的关系,函数本身也可以自调用,称为递归调用。

C程序的完整结构为:

```
外部变量说明;
main()
{
    内部变量说明;
    执行语句;
}
函数名(参量表)
参量说明;
{
    内部变量说明;
    执行语句;
}
```

为了以后各章举例方便,简单介绍一下输入和输出函数,详细介绍参见第三章。

输入函数调用形式:

scanf("转换格式控制串",变量名指针表);

其中,scanf()为输入库函数;双引号括起来的"转换格式控制串"常用的格式与下述输出相同;接收输入的变量要用其指针型,即变量名前加"取地址运算符&"。例如:

scanf("%d",&x);

用户从终端上打入的数(例如123),按十进制整数解释,赋给变量x。

输出函数调用形式:

printf("转换格式控制串",变量名表);

其中,printf()为输出库函数;双引号括起来的"转换格式控制串"有以下几种常用形式:

格式串	相应的参数输出形式
%d	十进制整数
%f	十进制浮点数
%c	单个字符
%s	字符串

用C++编写上述简单程序:

```
void main(void)                    // add of a and b
{       int  a,b,sum;
        a=123;
        b=456;
        sum=a+b;
        cout<<sum<<"\n";
```

}

与C编写的程序区别是：

(1)以“//”起始的行为注释行，以换行符结束，称为行注释符。

(2)void 表示空类型，当函数不带任何参量时必须加 void 说明，当函数不返回任何类型的数值时，为了避免误用，也要求加 void 说明。

(3)cout 是与标准输出设备相联，cout<<表示向标准输出设备输出，上例输出变量 sum 的值，在输出语句中不必加格式说明，因为 cout 会自动按照输出变量的格式输出。因此，在C++中不再使用 printf()函数，而改用“cout<<”。

如果上例中 a 和 b 是由标准输入终端输入的，则在C中通过以下语句实现：

```
scanf("%d %d",&a,&b);
```

在C++中通过以下语句实现：

```
cin>>a>>b;
```

其中，cin 与标准输入设备相联，cin>>表示从标准输入设备输入，因此可以代替 scanf()函数。

C 程序的文件名后缀为.c，例如，file.c。C++程序的文件名后缀为.cpp，例如，file.cpp。要用哪一个编译器编译，必须在文件名后加相应的后缀。

第二章　C语言基本概念(一)

2.1　基本数据类型

C语言中基本数据类型有四种:char(字符型),int(整数型),float(单精度浮点型)和double(双精度浮点型)。

1. **字符型(char)**

字符型数据有单个字符和字符串两种。

·单个字符占用一个字节(8位),通常用来表达字符集(ASCII码或EBCDIC码字符集)中的一个字符,字符常量是用单引号括起来的单个字符,'a','b'等;也可以用来表达短整数,取值范围为－128到＋127。若定义为无符号(unsigned)短整数,取值范围为0～255。

·字符串占用多个字节,要用字符型数组表达一个字符串,数组的每个元素(即为一个字节)存放串中一个字符,最后加入空白字符NULL(\0)来标志字符串结束。字符串常量用双引号括起来,如"abc","a"等。

注意区别'a'和"a",前者是单个字符a,后者是a和\0组成的字符串。

某些非图形字符可以用带反斜杠的扩展表示法,如表2.1所示。这些字符看上去好像是两个字符组成,但实际只起一个作用。它们在屏幕上显示不出来,一般起控制作用。注意区别'\0'和'0':'\0'表示具有编码值0的字符;而'0'表示的是字符0(在ASCII码中编码为48,在EBCDIC中为240)。\ddd为换码符,如'\014'表示ASCII码中的FF。

表2.1

非图形字符	扩展表示法	功　能
NL(LF)	\n	换　行
HT	\t	水平制表
BS	\b	退　格
CR	\r	回　车
FF	\f	走　纸
\	\\	反斜杠
'	\'	单引号
ddd	\ddd	三位八进制数

2. **整数型(int)**

这是表达整数值的数据形式,它的长度是所使用计算机的机器字长,如字长为16位,取值范围为－32768到＋32767。

整常数通常用十进制表示，也可以用八进制和十六进制表示。用八进制表示，开头要写上0(零)，其后再写上要表示的八进制数，八进制数用数码0～7表示。用十六进制表示时，开头要写上0x(或0X)，在十六进制的表示法里，除了数码0～9外，还使用字母A～F(或a～f)，对应十进制的10～15。

例如：　十进制数：31

八进制数：037

十六进制数：0x1f 或 0X1f

在整型数据中，根据数据的二进制位数多少有short(短整数)、int(基本整数)和long(长整数)之分，具体字长和具体的机器规定有关。

此外，还有无符号(unsigned)整型数据类型，也分为三类，即：unsigned short，unsigned int 和 unsigned long。描述无符号基本整型时，可以省略int，此时，取值范围为0～65535。

3. 单精度浮点型(float)

这是表达实数值的数据形式，由整数部分、小数部分和指数部分组成。占用两个字长(一般为32位)，小数点保留7位有效位，取值范围为-10^{38}到-10^{-38}；0；10^{-38}到10^{38}。

注意浮点数的表示方法，合法的表示如：

1.2e+3	(e代替10，即1.2×10^{3})；
1.2e−3	(相当于1.2×10^{-3})；
12.34	(缺省指数部分)；
.1234	(缺省整数和指数部分)。

以下是不合法表示：

4.5e	(e右边一定要有整数部分)；
.e+3	(小数点两边至少有一个数字部分)；
e−6	(整数部分和小数部分不能同时缺省)；
0.3e4.5	(指数部分要整数型数值)。

4. 双精度浮点型(double)

它是单精度浮点的两倍字长，一般为64位，十进制的有效位为15位，最大可表示到1e+308。

实际上所有浮点计算都是以double进行的，对于float变量先转换为double型(尾部添零)，然后运算，当结果赋给float变量时，对双精度结果进行截尾，截尾时四舍五入。

在C程序中，所有变量在使用之前，必须加以类型说明，通常放在函数开始处，在可执行语句之前。变量说明语句的格式为：

类型　　变量名表；

举例如下：

```
char   c;
int    lower,upper,step;
float  x,y;
double z;
unsigned int  i,j;
```

其中，c为字符型变量；lower、upper和step为整型变量；x、y为单精度浮点型变量；z为双精度浮点型变量；i、j为无符号整型变量。

C语言除了基本数据类型外，还可以由基本数据类型构造出复杂的数据形式，如数组就是其中一种。

数组由固定数量的同类型元素组成，即数组由若干个元素组成，且每一个元素具有相同的数据类型，并能用统一的数组名和下标唯一地确定数组中每一个元素。程序中使用数组时，首先要进行数组说明，编译时给所说明的数组分配一块连续存贮空间，一个元素存放一个数据。说明语句形式：

类型　　数组名[元素个数];

如：
```
int    a[5];
char   b[10];
float  c[10];
```

这三个说明语句，说明了数组a、b、c分别是整型、字符型和浮点型数组。其中a、b、c是数组名，方括号内的数字为元素个数，这些元素是从下标为0的元素开始，例如，a[5]的5个元素分别是：a[0]，a[1]，a[2]，a[3]，a[4]。每个元素占用2个字节，数组a总共占用10个字节。字符型数组一般用来存放字符串，一个元素存放一个字符(即占用一个字节)，b[10]共占用10个字节。浮点型数组c的每个元素占用两倍机器字长(即4个字节)，10个元素共占用40个字节。

C语言提供了二维数组，说明语句为：

类型　　数组名[行下标][列下标];

如：
```
int    a[10][20];
char   matrix[10][40];
```

这两个说明语句表示：数组a为10行20列整型数组；数组matrix为10行40列字符型数组。数组a共有200个元素，它们是按行存放的，a[0][0]，a[0][1]，…，a[0][19]，a[1][0]，…，a[1][19]，…，a[9][0]，…，a[9][19]；matrix共有400个元素，两者占用同样的存贮空间。

在C程序里，字符型、整型和浮点型数据可以在同一个表达式中混合使用。当两个操作数类型不一致时，C编译程序将按一定的规则自动进行类型转换。

转换规则可以归纳为以下原则：

把占用字节数较少的类型(称为较低类型)转换为占用字节数较多的类型(称为较高类型)。因此，有以下关系：

$$\text{char} < \text{int} < \text{unsigned} < \text{long} < \text{float} < \text{double}$$

左边为较低类型，右边为较高类型。当在一个表达式中具有不同类型的变量时，在运算前，首先把各个变量类型转换为表达式中出现的最高类型，当类型一致后，再进行运算。要注意的是，当表达式中最高类型为单精度浮点型(float)时，为了提高运算精度，所有变量类型都被转换为双精度浮点型(double)，运算结果也为double型。

类型转换也发生在赋值语句的左、右边。转换规则是：把赋值符右边的类型转换为左边变量的类型。此时，要注意精度的损失。例如，程序段：

```
int   i;
char  c;
i=c;
```

其中i为整型变量，c为字符变量。把c的值赋给变量i，此时，精度不损失。

如果赋值语句为

```
c=i;
```

将整型变量 i 的低 8 位赋给字符变量 c，丢掉了高 8 位数。使用时，必须特别注意。

又如：

```
int   i;
float   x;
i=x;
```

把浮点变量 x 值赋给整型变量 i，只取 x 的整数部分，舍去了 x 的全部小数部分。

除了机器自动按规则进行转换外，也可以使用 cast 算符强使显式的类型按用户要求进行转换。例如，x 为整型，要求它转换为双精度浮点型，通过(double)x 完成。一般库函数要求函数变量为 double 型，例如开平方根 sqrt(x)，要求 x 为 double 型，若 x 是某个表达式计算结果，不一定是 double 型，则可以强使转换，写为 sqrt((double)x)。

[例 2.1]

```
main ()
{     int   x,y;
      long   z;
      x=400;      y=200;
      z=(long)x * y;
      printf("%ld\n",z);
}
```

执行结果：　80000

其中，把 x 变量强使为 long 型，表达式中 y 自动转变为 long，运算后结果为 long 型，赋给变量 z。格式符%ld 表示按长整型输出变量 z 的值。

2.2　运算符

C 语言具有丰富的运算符。按其功能分为：算术运算符、逻辑运算符、关系运算符、位逻辑运算符、赋值运算符、递增和递减运算符以及其他运算符等。按其在表达式中操作对象的个数又可分为：一元运算符、二元运算符和三元运算符。

1. 算术运算符

算术运算符包括两种类型：一元算术运算符和二元算术运算符。

一元运算符仅有减法(取负)运算符：

　　-x　　把 x 乘以-1，其结果是取操作数负值；

二元运算符有五种：+(加)、-(减)、*(乘)、/(除)和%(取模或求余)。

表达式	含　义
x+y	将 x 与 y 相加；
x-y	x 减去 y；
x * y	x 乘以 y；
x/y	x 除以 y；
x%y	求 x 除以 y 的余数；

关于后两种运算符，需要指出两点：

(1)两个整数相除，只取其整数部分，零不能做除数。

下面举几个整数相除的例子：

7/4=1； 8/4=2； 4/5=0； -7/4=-1； 7/-4=-1；

(2)运算符%只适用于整数。余数的符号与被除数的符号相同，结果小于除数或为零。

如：

7%4=3； 8%4=0； 3%5=3； -7%4=-3； 7%-4=3；

2. 关系运算符和逻辑运算符

C 语言有 4 个关系运算符和 2 个等值运算符，即：>(大于)、>=(大于或等于)、<(小于)、<=(小于或等于)、==(等于)和!=(不等于)。

表达式	含 义
x>y	如果 x、y 的关系表达式成立，则为真，表达式值为 1。
x>=y	
x<y	如果 x、y 的关系表达式不成立，则为假，表达式值为 0。
x<=y	
x==y	
x!=y	

逻辑运算有 2 个二元运算符和 1 个一元运算符，即 &&(逻辑与)、||(逻辑或)和！(逻辑非)。

表达式	含 义
e1 && e2	当表达式 e1 和 e2 同时成立，则结果为真，否则为假。
e1 || e2	当表达式 e1 和 e2 之中某一个成立，结果为真；两个同时不成立，则结果为假。
! e1	如果 e1 为真，则结果为假；如果 e1 为假，则结果为真。

前两种表达式的运算顺序，必须是自左向右。对 && 算符，当 e1 为 0 时，则无须判断 e2，只有当 e1 不为 0 时，再继续判断 e2。而对 || 算符，当 e1 不为 0 时，则无须再判断 e2，只有当 e1 为 0 时，再继续判断 e2。若 e1 和 e2 不是关系表达式，则非零值为真，零值为假。

3. 位逻辑运算符

这是直接作用于各位的算符。所谓位运算，就是指进行二进制位的运算。在系统软件中常要处理二进位问题，C 提供了位运算符，可以实现由汇编语言所能完成的一些功能。位运算包括 5 种二元运算符和 1 种一元运算符。

(1)按位与(&)

两个操作数逐位求与，经常用于把特定位清 0。如：a=11011001，b=11110000

```
  11011001
& 11110000
----------   把 a 的低 4 位清为 0。
  11010000
```

这里所遵循的运算规则是：

0 & 0=0； 0 & 1=0；

1 & 0=0； 1 & 1=1；

(2)按位或(|)

两个操作数逐位相或，经常用于把特定位置成 1。如：a=11011001，b=00001111

```
  11011001
| 00001111
----------  把a的低4位置成1。
  11011111
```

这里所遵循的运算规则是：

0 | 0=0;　　0 | 1=1;

1 | 0=1;　　1 | 1=1;

(3)按位异或(^)

两个操作数按位相加，经常用于使特定位翻转，要使某位0翻转，只要使这个数与相应位为1的数进行^运算即可。如：a=11011001，b=00001111

```
  11011001
^ 00001111
----------  把a的低4位翻转
  11010110
```

所谓"异或"，就是当二位不同时才起"或"作用。

这里所遵循的运算规则是：

0 ^ 0=0;　　0 ^ 1=1;

1 ^ 0=1;　　1 ^ 1=0;

(4)左移算符(<<)和右移算符(>>)

将操作数左移所指定的位数，常用来使某个数扩大2的方幂倍，即右边空出来的位补上0。如一个八进制数左移2位，007777<<2得037774；

右移算符(>>)将操作数右移若干位。常用来把某数缩小2的方幂倍。分逻辑右移(即被移动的数为无符号数)，和算术右移(即被移动的数为有符号数)。若为逻辑右移，则在左边的空位上补上0；若为算术右移，则在左边的空位上补上原符号，这叫做符号扩张。

(5)反码运算符(~)

求操作数反码，即把操作数各位都取反。如

a=11011001

~a=00100110

4. 自增和自减运算符

自增和自减算符都是一元运算符，分别使变量值增1和减1。

++n　等价于　n=n+1;

--n　等价于　n=n-1;

在程序中，经常作为计数单元使用的指令。此外，这两种算符还有一种用法，即

n++和n--

也等价于n加1和n减1。区别在于：++n和--n是先把n加(或减)1，再使用；而n++和n--是先使用n的原有值，再进行n加(或减)1。如程序段：

n=3;　　m=++n;

执行结果　m=4，n=4。而m=n++；执行结果　m=3，n=4。

但不管是哪种情况，执行后的n值都是4。在循环流程和堆栈处理中都有效地使用了++和--算符。