C语言位运算

&与 |或 ^异或 ~取反 <<左移 >>右移

运算符的操作数只能是整型和字符型

与1与，原值；与0与，为0；

应用：（1）清零（2）判断某位是0还是1（3）

与0或，原值；与1或，为1；

异或，同0异1

应用：（1）不借助临时变量，交换两个整型变量的值

int a = 3;int b = 5;

a = a^b; b = b^a; a = a^b;

（2）特定位翻转

取反，01,10；

左移，左移n位，左弃n位，右补n个0；

【左移1位，相当于乘2；左移n位，乘2^n；

适用于左移时被溢出舍弃的高位中不包含1的情况，当左移位数超过位数，取余数，

例，8位00000010,左移6位，乘2^6，左移7位出错！左移8位，取余数0，相当于没左移，左移10位，相当于乘2^2】

右移，正数(无符号数，有符号正数)时，右移n位，右弃n位，左补n个0；

负数(有符号)时，右移n位，右弃n位，符号位为1，左补n-1个0或1，取决于编译器。

在C语言中还提供复合的位运算符，如下：

&=、!=、>>=、<<=和^=

例如：a&=0x11等价于 a= a&0x11，其他运算符以此类推。

不同类型的整数数据在进行混合类型的位运算时，按右端对齐原则进行处理，按数据长度大的数据进行处理，将数据长度小的数据左端补0或1。例如char a与int b进行位运算的时候，按int 进行处理，char a转化为整型数据，并在左端补0。

补位原则如下：

1) 对于有符号数据：如果a为正整数，则左端补0，如果a 为负数，则左端补1。

2) 对于无符号数据：在左端补0。

位运算的应用

（1）判断二进制数从最低位起第i位是0还是1

将1左移i-1位与该数，结果是0，则第i位是0，否则是1

（2）判断一个整数的奇偶（...-2,-1,0,1,2...）

int n;

if( 1 == n & 1 ) n是奇数

if( 0 == n & 1 ) n是偶数

原理：

1. 奇数+奇数等于偶数，偶数+偶数等于偶数，奇数+偶数等于奇数
2. 二进制整数，除尾数外，其他权位必定是偶数，尾数是0，偶+偶=偶；尾数是1，偶+奇=奇
3. n & 1，除尾数其他位都是0，尾数是0，十进制结果为0，尾数是1，十进制结果是1
4. 简而言之，二进制整数，尾数是0，是偶数，尾数是1，是奇数。