位运算应用口诀和实例

位运算应用口诀：

清零取反要用与，某位置一可用或

若要取反和交换，轻轻松松用异或

移位运算：

要点

1．它们都是双目运算符，两个运算分量都是整型，结果也是整型。

2．" < <" 左移：右边空出的位上补 0，左边的位将从字头挤掉，其值相当于乘 2。

3．">>"右移：右边的位被挤掉。对于左边移出的空位，如果是正数则空位补 0，若为负数，可能补 0 或补 1，这取决于所用的计算机系统。

4． ">>>"运算符，右边的位被挤掉，对于左边移出的空位一概补上 0。

位运算符的应用 (源操作数 s 掩码 mask)

按位与-- &

1．清零特定位 (mask 中特定位置 0，其它位为 1，s=s&mask)

2．取某数中指定位 (mask 中特定位置 1，其它位为 0，s=s&mask)

按位或-- |

常用来将源操作数某些位置 1，其它位不变。 (mask 中特定位置 1，其它位为 0 s=s|mask)

位异或-- ^

1．使特定位的值取反 (mask 中特定位置 1，其它位为 0 s=s^mask)

2．不引入第三变量，交换两个变量的值 (设 a=a1, b=b1)

目 标　　操 作　　操作后状态

a=a1^b1　　a=a^b　　a=a1^b1,b=b1

b=a1^b1^b1　　b=a^b　　a=a1^b1,b=a1

a=b1^a1^a1　　a=a^b　　a=b1,b=a1

二进制补码运算公式：

-x = ~x + 1 = ~(x-1)　　~x = -x-1

-(~x) = x+1　　~(-x) = x-1

x+y = x - ~y - 1 = (x|y)+(x&y)　　x-y = x + ~y + 1 = (x|~y)-(~x&y)

x^y = (x|y)-(x&y)　　x|y = (x&~y)+y

x&y = (~x|y)-~x　　x==y:　　~(x-y|y-x)

x!=y:　　x-y|y-x

x < y:　　(x-y)^((x^y)&((x-y)^x))

x <=y:　　(x|~y)&((x^y)|~(y-x))

x < y:　　(~x&y)|((~x|y)&(x-y))//无符号 x, y 比较

x <=y:　　(~x|y)&((x^y)|~(y-x))//无符号 x, y 比较

应用举例

(1) 判断 int 型变量 a 是奇数还是偶数

★ a&1　= 0 偶数

★ a&1 =　1 奇数

(2) 取 int 型变量 a 的第 k 位 (k=0,1,2……sizeof(int))，即 a>>k&1

(3) 将 int 型变量 a 的第 k 位清 0，即 a=a&~(1 < <k)

(4) 将 int 型变量 a 的第 k 位置 1，　即 a=a|(1 < <k)

(5) int 型变量循环左移 k 次，即 a=a < <k|a>>16-k　(设 sizeof(int)=16)

(6) int 型变量 a 循环右移 k 次，即 a=a>>k|a < <16-k　(设 sizeof(int)=16)

(7) 整数的平均值

对于两个整数 x,y，如果用 (x+y)/2 求平均值，会产生溢出，因为 x+y 可能会大于 INT_MAX，但是我们知道它们的平均值是肯定不会溢出的，我们用如下算法：

```
int average(int x, int y)   //返回 X,Y 的平均值
{       return (x&y)+((x^y)>>1);          }
```

(8)判断一个整数是不是 2 的幂,对于一个数 x >= 0，判断他是不是 2 的幂

```
boolean power2(int x)
{       return ((x&(x-1))==0)&&(x!=0);       }
```

(9)不用 temp 交换两个整数

```
void swap(int x , int y)
{       x ^= y;              y ^= x;              x ^= y;      }
```

(10)计算绝对值

```
int abs( int x )
{
      int y ;
      y = x >> 31 ;
      return (x^y)-y ;            //or: (x+y)^y
}
```

(11)取模运算转化成位运算 (在不产生溢出的情况下)

a % (2^n) 等价于 a & (2^n - 1)

(12)乘法运算转化成位运算 (在不产生溢出的情况下)

a * (2^n) 等价于 a < < n

(13)除法运算转化成位运算 (在不产生溢出的情况下)

a / (2^n) 等价于 a>> n

例: 12/8 == 12>>3

(14) a % 2 等价于 a & 1

(15) if (x == a) x= b;

else x= a;

等价于 x= a ^ b ^ x;

(16) x 的 相反数 表示为 (~x+1)

实例

功能	示例	位运算
去掉最后一位	(101101->10110)	x >> 1
在最后加一个 0	(101101->1011010)	x < < 1
在最后加一个 1	(101101->1011011)	x < < 1+1
把最后一位变成 1	(101100->101101)	x | 1
把最后一位变成 0	(101101->101100)	x | 1-1
最后一位取反	(101101->101100)	x ^ 1
把右数第 k 位变成 1	(101001->101101,k=3)	x | (1 < < (k-1))
把右数第 k 位变成 0	(101101->101001,k=3)	x & ~ (1 < < (k-1))
右数第 k 位取反	(101001->101101,k=3)	x ^ (1 < < (k-1))
取末三位	(1101101->101)	x & 7
取末 k 位	(1101101->1101,k=5)	x & ((1 < < k)-1)
取右数第 k 位	(1101101->1,k=4)	x >> (k-1) & 1

把末 k 位变成 1　　| (101001->101111,k=4)　　| x | (1 < < k-1)

末 k 位取反　　| (101001->100110,k=4)　　| x ^ (1 < < k-1)

把右边连续的 1 变成 0　　| (100101111->100100000)　　| x & (x+1)

把右起第一个 0 变成 1　　| (100101111->100111111)　　| x | (x+1)

把右边连续的 0 变成 1　　| (11011000->11011111)　　| x | (x-1)

取右边连续的 1　　| (100101111->1111)　　| (x ^ (x+1)) >> 1

去掉右起第一个 1 的左边　| (100101000->1000)　　| x & (x ^ (x-1))

判断奇数　　(x&1)==1

判断偶数　(x&1)==0

说说异或运算^和他的一个常用作用

作者：AOL　　类型:原创　　来源:

说说异或运算^和他的一个常用作用。

异或的运算方法是一个二进制运算：

1^1=0

0^0=0

1^0=1

0^1=1

两者相等为 0,不等为 1.

这样我们发现交换两个整数的值时可以不用第三个参数。

如 a=11,b=9.以下是二进制

a=a^b=1011^1001=0010;

b=b^a=1001^0010=1011;

a=a^b=0010^1011=1001;

这样一来 a=9,b=13 了。

举一个运用， 按一个按钮交换两个 mc 的位置可以这样。

```
mybt.onPress=function()
{
mc1._x=mc1._x^mc2._x;
mc2._x=mc2._x^mc1._x;
mc1._x=mc1._x^mc2._x;
//
mc1._y=mc1._y^mc2._y;
mc2._y=mc2._y^mc1._y;
mc1._y=mc1._y^mc2._y;
}
```

这样就可以不通过监时变量来传递了。