二进制变换操作

功能	示例	位运算
去掉最后一位	(101101->10110)	x shr 1
在最后加一个 0	(101101->1011010)	x shl 1
在最后加一个 1	(101101->1011011)	x shl 1+1
把最后一位变成 1	(101100->101101)	x or 1
把最后一位变成 0	(101101->101100)	x or 1-1
最后一位取反	(101101->101100)	x xor 1
把右数第 k 位变成 1	(101001->101101,k=3)	x or (1 shl (k-1))
把右数第 k 位变成 0	(101101->101001,k=3)	x and not (1 shl (k-1))
右数第 k 位取反	(101001->101101,k=3)	x xor (1 shl (k-1))
取末三位	(1101101->101)	x and 7
取末 k 位	(1101101->1101,k=5)	x and (1 shl k-1)
取右数第 k 位	(1101101->1,k=4)	x shr (k-1) and 1
把末 k 位变成 1	(101001->101111,k=4)	x or (1 shl k-1)
末 k 位取反	(101001->100110,k=4)	x xor (1 shl k-1)
把右边连续的 1 变成 0	(100101111->100100000)	x and (x+1)
把右起第一个 0 变成 1	(100101111->100111111)	x or (x+1)
把右边连续的 0 变成 1	(11011000->11011111)	x or (x-1)
取右边连续的 1	(100101111->1111)	(x xor (x+1)) shr 1
去掉右起第一个 1 的左边	(100101000->1000)	x and (x xor (x-1))

二进制中的1有奇数个还是偶数个

```
var
  x:longint;
begin
  readln(x);
  x:=x xor (x shr 1);
  x:=x xor (x shr 2);
  x:=x xor (x shr 4);
  x:=x xor (x shr 8);
  x:=x xor (x shr 16);
  writeln(x and 1);
end.
```

奇数 = 1 偶数 = 0

计算二进制的1的个数

```
x := (x and $55555555) + ((x shr 1) and $55555555);
x := (x and $33333333) + ((x shr 2) and $33333333);
x := (x and $0F0F0F0F) + ((x shr 4) and $0F0F0F0F);
x := (x and $00FF00FF) + ((x shr 8) and $00FF00FF);
x := (x and $0000FFFF) + ((x shr 16) and $0000FFFF);
```

1	1	0	1	0	0	1	1	<---原数
1 0		0 1		0 0		1 0		<---第一次运算后
0 0 1 1				0 0 1 0				<---第二次运算后
0 0 0 0 0 1 0 1								<---第三次运算后，得数为 5

整个程序是一个分治的思想。第一次我们把每相邻的两位加起来，得到每两位里 1 的个数，比如前两位 10 就表示原数的前两位有 2 个 1。第二次我们继续两两相加，10+01=11，00+10=10，得到的结果是 00110010，它表示原数前 4 位有 3 个 1，末 4 位有 2 个 1。最后一次我们把 0011 和 0010 加起来，得到的就是整个二进制中 1 的个数。程序中巧妙地使用取位和右移，比如第二行中$33333333 的二进制为 00110011001100....，用它和 x 做 and 运算就相当于以 2 为单位间隔取数。shr 的作用就是让加法运算的相同数位对齐。

二分查找32位整数的前导0个数

```
int nlz(unsigned x)
{
  int n;
  if (x == 0) return(32);
  n = 1;
  if ((x >> 16) == 0) {n = n +16; x = x <<16;}
  if ((x >> 24) == 0) {n = n + 8; x = x << 8;}
  if ((x >> 28) == 0) {n = n + 4; x = x << 4;}
  if ((x >> 30) == 0) {n = n + 2; x = x << 2;}
  n = n - (x >> 31);
  return n;
}
```

二进制逆序

```
var
  x:dword;
begin
  readln(x);
  x := (x and $55555555) shl 1 or (x and $AAAAAAAA) shr 1;
  x := (x and $33333333) shl 2 or (x and $CCCCCCCC) shr 2;
  x := (x and $0F0F0F0F) shl 4 or (x and $F0F0F0F0) shr 4;
  x := (x and $00FF00FF) shl 8 or (x and $FF00FF00) shr 8;
  x := (x and $0000FFFF) shl 16 or (x and $FFFF0000) shr 16;
  writeln(x);
end.
```

Gray码

第 i 位Gray码 = i xor (i>>1)