西北工业大学
NORTHWESTERN POLYTECHNICAL UNIVERSITY

C程序设计

Programming in C

1011014

主讲：姜学锋，计算机学院

数值数据的运算与处理

- 3、自增自减运算
- 4、位运算

2.4.3 自增自减运算符

表2-4 自增自减运算符

运算符	功能	目	结合性	用法
++	后置自增	单目	自右向左	lvalue++
--	后置自减	单目	自右向左	lvalue--
++	前置自增	单目	自右向左	++lvalue
--	前置自减	单目	自右向左	--lvalue

2.4.3 自增自减运算符

- 示例

```
int m=4, n;
① n = ++m;
  //m先增1，m为5，然后表达式使用m的值，赋值给n，n为5
② n = --m;
  //m先减1，m为4，然后表达式使用m的值，赋值给n，n为4
③ n = m++;
  //表达式先使用m的值，赋值给n，n为4，然后m增1，m为5
④ n = m--;
  //表达式先使用m的值，赋值给n，n为4，然后m减1，m为3
```

2.4.3 自增自减运算符

- 示例

```
int n=4 , m=4;
n++; //运算后n为5
++m; //运算后m为5
const int k=6;
5++;          //错误
--(a+b);      //错误
k++;          //错误
max(a,b)--; //错误
```

2.4.3 自增自减运算符

- 当一个表达式中对同一个变量进行多次自增自减运算，例如：

```
k=i+++i+++i++          //写法不直观，应写成k=(i++)+(i++)+(i++)
k=(i++)+(++i)+(++i) //可读性差，难于明确求值顺序
```

2.4.3 自增自减运算符

- 实际编程中，一个表达式中尽量不要出现过多的++或—运算，如果需要可以拆成多个表达式来写。

```
k=(i++)+(++i)+(++i) //可读性差，难于明确求值顺序
//如下写法更好
++i;
++i;
k=i;
i++;
```

2.4.7 位运算符

表2-9 位运算符

运算符	功能	目	结合性	用法
~	按位取反	单目	自右向左	~expr
<<	按位左移	双目	自左向右	expr1 << expr2
>>	按位右移	双目	自左向右	expr1 >> expr2
&	按位与	双目	自左向右	expr1 & expr2
^	按位异或	双目	自左向右	expr1 ^ expr2
|	按位或	双目	自左向右	expr1 | expr2

2.4.7 位运算符

- 由于是按二进制位进行运算，所以位运算符的运算对象应是整型数据。如果是有符号整型，位运算操作会连同符号位一起执行，使得符号发生不正常的变化。所以位运算符一般使用无符号整型，如字节（unsigned char）、字（unsigned short）、双字（unsigned int）等。

2.4.7 位运算符

- 1．按位与运算符（&）
- 参加运算的两个数据按二进制位进行与运算，运算规则是：
  - 0 & 0 = 0，0 & 1 = 0，1 & 0 = 0，1 & 1 = 1
  - 0 & X = 0，1 & X = X
- 例如，7&9的结果是1，运算过程为：

```
   00000111    ……  7
&  00001001    ……  9
-------------------
   00000001    ……  1
```

2.4.7 位运算符

- 如果参加运算的是负数，则以补码形式进行位与运算。例如，-7&-9的结果是-15，运算过程为：

```
  11111001    ……  -7
& 11110111    ……  -9
-----------------------
  11110001    ……  -15
```

2.4.7 位运算符

- 按位与和逻辑与是不同的。例如：
- 1&2按位与的结果是0，而1&&2逻辑与的结果是1（真）；4&7按位与的结果是4，而4&&7逻辑与的结果是1（真）。

2.4.7 位运算符

- 按位与能够实现一些特殊要求的运算：
- （1）指定二进制位清零。

```
    10101010    ……  170
&   11100111    ……  231
-------------------------
    10100010    ……  162
```

2.4.7 位运算符

- （2）取整数中指定二进制位。

```
      1010 1010    ……  170
&     0000 1111    ……   15
-------------------------
      0000 1010    ……   10
```

2.4.7 位运算符

- （3）保留指定位。

```
      10101010    ……  170
&     10000000    ……  128
-------------------------
      10000000    ……  128
```

2.4.7 位运算符

- 2．按位或运算符（|）
- 参加运算的两个数据按二进制位进行或运算，运算规则是：
  - 0|0 = 0，0|1 = 1，1|0 = 0，1|1 = 1
  - 1|X = 1，0|X = X
- 例如，7|9的结果是15，运算过程为：

```
  00000111    …… 7
| 00001001    …… 9
--------------------
  00001111    …… 15
```

2.4.7 位运算符

- 按位或和逻辑或是不同的。例如：
- 1|2按位或的结果是3，而1||2逻辑或的结果是1（真）
- 4|7按位或的结果是7，而4||7逻辑或的结果是1（真）。

2.4.7 位运算符

- 按位或常用来设置一个整数中指定二进制位为1。

```
  10101010   ……  170
| 01000000   ……   64
----------------------
  11101010   ……  234
```

2.4.7 位运算符

- 3．按位异或运算符（^）
- 参加运算的两个数据按二进制位进行异或运算，所谓异或是指两个二进制数相同为0，相异为1，运算规则是：
  - 0 ^ 0 = 0，0 ^ 1 = 1，1 ^ 0 = 1，1 ^ 1 = 0
- 例如，7^9的结果是14，运算过程为：

```
    00000111    …… 7
^   00001001    …… 9
-------------------------
    00001110    …… 14
```

2.4.7 位运算符

- 特别地，a&b+a^b等于a|b。

- 假设某二进制位为X（0或1），显然
- 0 ^ X = X，1 ^ X = $\bar{X}$，X ^ X = 0

- $\bar{X}$ 是X的翻转，即当X是0时，$\bar{X}$ 是1，当X是1时，$\bar{X}$ 是0。

2.4.7 位运算符

- 异或的作用是判断两个对应的位是否为异，按位异或能够实现一些特殊要求的运算：
- （1）使指定位翻转。

```
  10101010    ……  170
^ 00001111    ……   15
------------------------
  10100101    ……  165
```

2.4.7 位运算符

- （2）将两个值互换。

```
a = a ^ b , b = b ^ a , a = a ^ b ;
```

  - ①前两个表达式执行后，b = b ^ (a ^ b)，而b ^ (a ^ b)等于a ^ b ^ b，因此b的值等于a ^ 0，即a；
  - ②由于a = a ^ b，b = b ^ a ^ b，第3个表达式执行后，a = (a ^ b) ^ (b ^ a ^ b)，即a的值等于a ^ a ^ b ^ b ^ b，等于b。

2.4.7 位运算符

- 4．按位取反运算符（~）
- （~）是单目运算符，将一个整数中所有二进制位按位取反，即0变1，1变0。
- 例如，7按位取反的结果是248，运算过程为：

$$
\begin{array}{ll}
 & 00000111 \quad \cdots\cdots \quad 7 \\
\sim & \downarrow \\
\hline
 & 11111000 \quad \cdots\cdots 248
\end{array}
$$

2.4.7 位运算符

- 无论是何种整型数据，-1按补码形式存储时二进制位全是1，因此一个整数m与~m的加法或者按位或结果必然是-1，即：m + ~m=-1，m | ~m=-1。

2.4.7 位运算符

- 5．左移运算符（<<）
- （expr1 << expr2）的作用是将expr1的所有二进制位向左移expr2位，左边expr2位被移除，右边补expr2位0。
- 例如整数7（00000111）左移两位，即7<<2的结果是28（00011100）。

2.4.7 位运算符

- 6．右移运算符（>>）
- （expr1 >> expr2）的作用是将expr1的所有二进制位向右移expr2位，右边expr2位被移除，左边补expr2位0。
- 例如整数9（00001001）右移两位，即9>>2的结果是2（0000010）。

2.4.7 位运算符

【例2.8】

取一个字节型整数的D3～D5位。

解：令整数为m，其二进制位形式为XXnnnXXX，nnn为D3～D5位，X为其余位。

首先使m右移3位，将nnn移到低位，m>>3的结果是000XXnnn。

其次将结果与7（00000111）按位与，(m>>3) & 7的结果是00000nnn，得到m的D3～D5位。

2.4.7 位运算符

【例2.9】

将一个字节型整数m循环移动n位。

解：循环移位是指将m向右或向左移动n位，移除的位放在m的左边或右边。如 XXXXXXnn 循环右移2位的结果是 nnXXXXXX，nnnXXXXX 循环左移3位的结果是 XXXXXnnn。

2.4.7 位运算符

【例2.9】

（1）m循环右移n位

首先m右移n位，如XXXXXXnn右移2位得到00XXXXXX，然后m左移8-n位，如XXXXXXnn左移8-2位得到nn000000，将两个结果按位或即得到nnXXXXXX，表达式为：m>>n | m<<8-n

2.4.7 位运算符

【例2.9】

（2）m循环左移n位

首先m左移n位，如nnnXXXXX左移3位得到XXXXX000，然后m右移8-n位，如nnnXXXXX右移8-3位得到00000nnn，将两个结果按位或即得到XXXXXnnn，表达式为：m<<n | m>>8-n


程序设计