**一、操作符分类：**

**算数操作符： + - \* / %**

整数的除法： 1/2 ---> 0

浮点数的除法：1.0 / 2 ---> 0.5

% 计算的是整数之后的余数 - 取模操作符的两端必须是整数

**移位操作符：<< >> 二进制**

<< 左移操作符

>> 右移操作符

移位操作符，移动的是二进制位（补码）

整数的二进制表示有3种

原码、反码、补码

正的整数的原码、反码、补码相同

负的整数的原码、反码、补码是要计算的

例：7的二进制111

00000000000000000000000000000111 - 原码、反码、补码

-7

10000000000000000000000000000111 - 原码

11111111111111111111111111111000 - 反码（原码的符号位不变，其他位按位取反就是反码）

11111111111111111111111111111001 - 补码（反码+1就是补码）

整数在内存中存储的是补码

移位操作符移动的是其二进制的补码

左移操作符：

左边丢弃，右边补0 7：00000000000000000000000000000111 ---> 00000000000000000000000000001110

-7：11111111111111111111111111111001 ---> 11111111111111111111111111110010补码

---> 11111111111111111111111111110010 反码

---> 10000000000000000000000000001110 原码

左移操作符有乘2的效果

右移操作符：

1、算术移位：右边丢弃，左边补原符号位

2、逻辑移位：右边丢弃，左边补0

VS编译器采用的是算数右移

注：

左右移操作符只能对整数使用

移位操作符不能移动负数位，标准未定义

int main()

{

    int a = 7;

    //左移操作符

    int b = a << 1;

    printf("a=%d\n", a);    //7

    printf("b=%d\n", b);    //14

}

int main()

{

    int a = -7;

    //左移操作符

    int b = a << 1;

    printf("a=%d\n", a);    //-7

    printf("b=%d\n", b);    //-14

}

int main()

{

    int a = 7;

    //左移操作符

    int b = a >> 1;

    printf("a=%d\n", a);    // 7

    printf("b=%d\n", b);    // 3

}

int main()

{

    int a = -7;

    //左移操作符

    int b = a >> 1;

    printf("a=%d\n", a);    // -7

    printf("b=%d\n", b);    // -4

}

**位操作符：& | ^**

**赋值操作符：= += -=**

**单目操作符：**

**关系操作符**

**逻辑操作符**

**条件操作符**

**逗号操作符**

**下标操作符、函数调用和结构成员**