-20　原1001 0100　反1110 1011 补1110 1100

10　0000 1010　**1111 0110**-->1111 0101--->1000 1010 -10

1000 0000 0001 1110

补充：

有符号数和无符号数一起参与运算，会发生隐式强转，结果是一个无符号数。

```
#include <stdio.h>
int main(int argc, const char *argv[])
{
    unsigned int a = 10;
    signed int b = -20;
    if (a+b>0) //无符号数和有符号数一起参与运算，
        //发生了隐式强转,把有符号数强转成无符号数所以a+b结果>0
    {
        printf("yes\n");
    }
    printf("%d\n",a+b);//%d，又把无符号数强转成有符号的十进制进行输出
    printf("%u\n",a+b);
    return 0;
}
```

一、运算符

算术运算符：+ - * / %（模除）自增自减

关系运算符：> < >= <= == !=

逻辑运算符：&& || !

位运算符：~ << >> & | ^

赋值运算符：+= -= /= *=

条件运算符：C语言中唯一一个三目运算符

逗号运算符：(，，.....);

sizeof运算符：计算数据占用空间的大小，以字节为单位

1.1算术运算符

/ 除数不能为0；

表达一1/表达式2：如果两个表达式都为int，结果就是整除的值，如果有一个浮点型，结果就是浮点型。

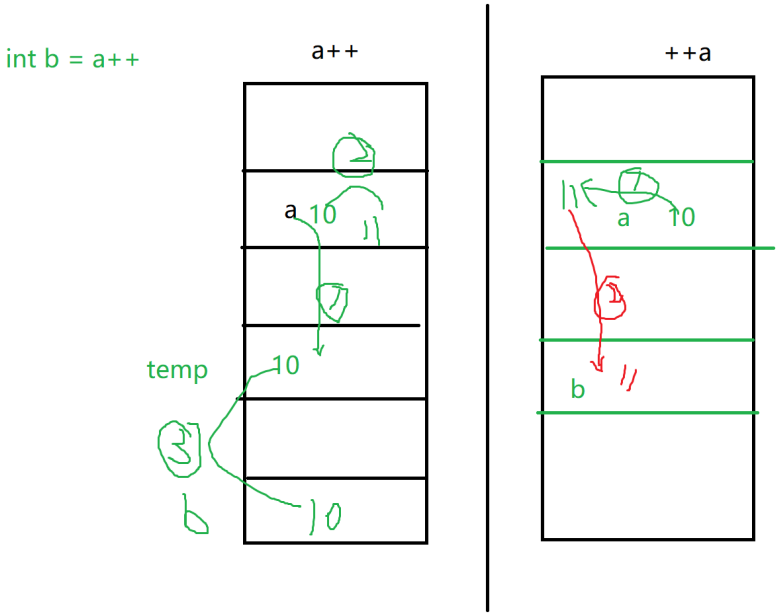
%（模除）**左右两个表达式必须是整形**

模除是取两个整形相除的余数

1.2自增自减

a++ 、 ++a

这个两个a都加1了，但是两个表达式的结果不同。



--a，a--同理。

简单的记法：

a++，是先看见变量，后看见运算符，所以先用变量的值，再运算。

++a，是先看见运算符，后看见变量，所以先运算，再使用变量。

1.3关系运算符

关系运算的结果是0/1，结果为真1，结果为假0　非0为真，0为假

<= 小于等于　　>=大于等于　　==判读那两个数相等　！=两个数不相等

1.4逻辑运算符

逻辑与&&：逻辑与两侧的表达式全为真，结果才是真。

中秋节放假&&买到火车票&&没有疫情---->中秋节回家

逻辑或||：逻辑或两侧的表达式，有一个为真结果就是真。

逻辑短路现象：

逻辑与左侧的表达式为假时，结果就是0，不会执行右侧的表达式

逻辑或左侧的表达式为真时，结果就是1，不会执行右侧的表达式

```c
#include <stdio.h>
int main(int argc, const char *argv[])
{
    int a = 10;
    int b =14;
    int c ;
    c = a>b&&a++; //c的值是0，逻辑与，左侧表达式为假，发生了逻辑短路现象，a不会自增
    c = a<b||a++; //c的值是1，逻辑或，左侧表达式为真，发生了逻辑短路现象，a不会自增
    printf("%d\n",a); //10
    printf("%d\n",c); //0
    return 0;
}
```

1.5位运算符

都是对二进制数进行运算的，

~：按位取反

char a = 1000 1010;

~a = 0111 0101;

10 ： 0000 1010　　0000 0010

1111

<<：按位左移,高位截断，低位补0

10<<2:10按位左移2

0000 1010<<2 =0010 1000

0100 1011 把他1，4位置0，3，6位置1

```
>>:按位右移，低位截取高位补0   10<<2 10按位右移2
^:按位异或，同0异1
&：按位与，一个0就是0
|：按位或，一个1就是1
```

10 ： 0000 1010　2：0000 0010

0000 1000

按位或：0000 1010

按位与：0000 0010

19:0001 0011　　2:0000 0010

19^2 ：0001 0001

19&2 ：0000 0010

19|2 ：0001 0011

口诀：0与任何数，都是0**

0或任何数，都是数本身

1与任何数，都是数本身

1或任何数，都是1

练习：0100 1011，把这个数的第1，4位置0，第3，6置1。从右到左

0110 0110

按位与，再按位或：1）&1111 0110 ---> 0100 0010

2) | 0010 0100----> 0110 0110

异或：0010 1101 ---> 0110 0110

1.6赋值运算符

左值 = 右值;

把右值赋给左值，赋值运算符的左值必须是一个变量。

+=： a+=1； ----> a= a + 1;

-= : a-=1; ---->a =a -1;

*=:

1.7条件运算符

格式：表达式1？表达式2：表达式3;

运算逻辑：表达式1为真，输出表达式2，否则就是表达式3.

1.8逗号运算符

格式：(表达式1，表达式2，表达式3.....);

运算逻辑：从左向右执行，取最后一个表达式的结果，其他也会执行。

1.9sizeof运算符

计算数据的大小，以字节为单位，默认结果是无符号的长整型。

sizeof(数据);

1.10运算符的优先级

单目运算符>算术运算符>左移右移>关系运算符>位运算符>逻辑运算符>条件运算符>赋值运算符>逗号运算符

在运算过程中，如果我们掌握不好运算符的优先级顺序，就加括号

优先级	运算符及其含义	结合规律
1	[ ] ( ) . -> 后缀++ 后缀--	从左向右
2	前缀++ 前缀-- sizeof & * + - ~ !	从右向左
3	强制类型转换	从右向左
4	* / % (算术乘除)	从左向右
5	+ - (算术加减)	从左向右
6	<< >>（位移位）	从左向右
7	< <= > >=	从左向右
8	== !=	从左向右
9	& (位逻辑与)	从左向右
10	^ (位逻辑异或)	从左向右
11	| (位逻辑或)	从左向右
12	&&	从左向右
13	||	从左向右
14	? :	从右向左
15	= *= /= %= += -= <<= >>= &= ^= |=	从右向左
16	,	从左向右

二、输入输出函数

2.1标准输出函数

printf("");

printf," "里面的内容，除了放占位符和转义字符，其他的都原封不动的输出。

函数原型：

```
int printf(const char *restrict, ...);
```

使用：printf("占位符"，输出表象);

占位符

```
%d    以有符号的十进制数输出整形
%l    输出长整型或双精度浮点型
%nd   以指定宽度n输出整形数据，宽度不足补0，右对齐
%-nd  以指定宽度n输出整型数组，宽度不足补0，左对齐
%.mf  输出小数点后m位，超出部分，四舍五入
%f    输出浮点型
%o    以八进制格式输出
%x    以十六进制格式输出
%#    输出对应进制的前导符
%%    输出%
%c    输出单个字符
%s    输出字符串
```

2.2标准输入函数

scanf("占位符"，占位符对应的数据类型的变量的**地址**);

sacnf,一次只能识别到空格，tab键和\n之前，是严格控制格式的函数，双引号内的格式是什么，输入的格式就是什么。

scanf吸收垃圾字符的三种方式

```c
#include <stdio.h>
int main(int argc, const char *argv[])
{
    char a,b,c,d;
    //第一种方式，用getchar吸收
    scanf("%c",&a);
    getchar();
    scanf("%c",&b);
    getchar();
    scanf("%c",&c);
    getchar();
    scanf("%c",&d);

    //第二种方法，在%c前面家空格。
    scanf(" %c",&a);
    putchar(a);

    //第三种办法，利用scanf严格控制格式的特点。
    scanf("%c ",&a);
    scanf("%c ",&b);
    scanf("%c ",&c);
    scanf("%c",&d);

    //第四种办法：通过抑制字符吸收，不推荐使用，因为抑制字符一定要吃掉一个字符，可能会吃掉有用的字符
    scanf("%c%*c%c%*c%c",&b,&c,&d);

    printf("%c\n",a);
    printf("%c\n",b);
    printf("%c\n",c);
    printf("%c\n",d);
    printf("%c\n",e);

    /*scanf("%d,%d,%d,%d",&b,&a,&d,&e);
    //printf("%c\n",c);
```

```
    printf("%d\n",b);
    printf("%d\n",d);
    printf("%d\n",a);
    printf("%d\n",e);*/

    return 0;
}
```

2.3getchar

从终端获取一个字符，多用于scanf吸收垃圾字符。

2.4putchar

输出单个字符。

格式： putchar();

括号里面可以放字符，也可以放char类型的变量还可以放字符的ASCII码。

三、分支控制语句

C语言的结构

1）顺序结构

2）分支结构

3）循环结构

3.1if…else语句

if语句

```
if(表达式1)
{
    表达式1成立，执行的语句;      //代码块
}
```

if….else语句

```
if(表达式1)
{
    表达式1成立，执行的语句;      //代码块
}
else
{
    表达式1不成立时，执行的语句;
}
```

else必须匹配一个if，但是if可以单独存在