# 目录

[目录 1](#_Toc519093902)

[第2章 运算符&表达式 2](#_Toc519093903)

[2.1 运算符与表达式概述 2](#_Toc519093904)

[2.2 运算符 2](#_Toc519093905)

[2.2.1 赋值运算符 2](#_Toc519093906)

[2.2.2 算术运算符 2](#_Toc519093907)

[2.2.3 关系运算符 2](#_Toc519093908)

[2.2.4 位运算符 2](#_Toc519093909)

[2.2.5 逻辑运算符 2](#_Toc519093910)

[2.2.6 复合运算符 2](#_Toc519093911)

[2.2.7 条件运算符 2](#_Toc519093912)

[2.2.8 逗号运算符 2](#_Toc519093913)

# 运算符&表达式

## 运算符与表达式概述

1. 运算符：对数据的一种操作规则，或者说按照一定规则来操作数据 1+2 3\*4
2. 操作数：运算符操作的对象
3. 表达式：运算符和操作数组合的式子 1+2(算术表达式) 1(常量表达式) a(变量表达式)
4. 优先级：不同运算符同时出现的时候，运算的顺序 1 + 2 \* 3 (1 + 3) / 2
5. 结合性：不同或者相同运算符同时出现的时候，但是优先级一样，运算顺序 1+2-4 \*a++

注意：只有优先级一样，才需要去考虑结合性

优先级不一样，不需要考虑结合性

1. 根据操作数的个数
2. 单目运算符：操作数的个数为1 a++ a-- &a \*p
3. 双目运算符：操作数的个数为2 + - \* / ….
4. 三目运算符：操作数的个数为3 唯一三目的就是条件运算符?:

## 运算符

### 赋值运算符

1 符号：=(赋值) ==(等于)

2 赋值表达式：变量 = 表达式(常量、变量、表达式) a = 10; a = b; a = b + 10;

3 说明：赋值运算符左边必须为变量(空间)，不可以为常量、表达式

4 特殊用法

一般情况下，都是同类型数据进行计算

常见的，将浮点型数据赋值给整型变量，数据会丢失，取整(不符合四舍五入)

将整型数据赋值给浮点型变量，数据不会丢失，添加小数点后面6个0

5 举例(单片机用的到)

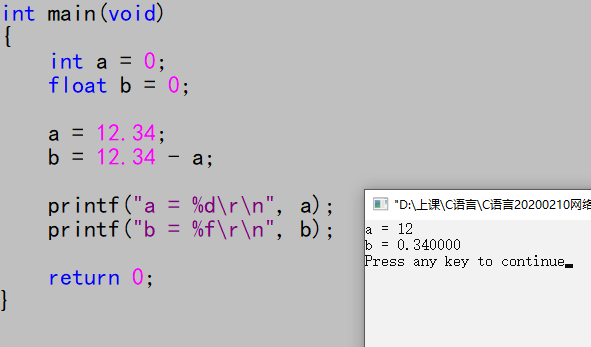
取出12.34这个浮点型数据整数部分和小数部分

int a;//存整数

float b;//存小数

a = 12.34;// 浮点型数据赋值给整型变量 取整

b = 12.34 – a;//得到小数



6 数据类型转换

1. 自动数据类型转换：不需要人为干预

数据进行运算，必须是在相同类型的情况下进行的

空间小🡪空间大、低精度🡪高精度

char、short 🡪 int 🡪 long 🡪 float 🡪 double

char a = 1;

int b = 12;

float c = 3.45;

double d = 12.34;

d + a \* b + c ?都为转化为什么类型进行运算？

d + (double)((int)a \* b) + (double)c 🡪 double

1. 强制数据类型转换：需要人来干预

格式：(数据类型) 表达式

char a = 1;

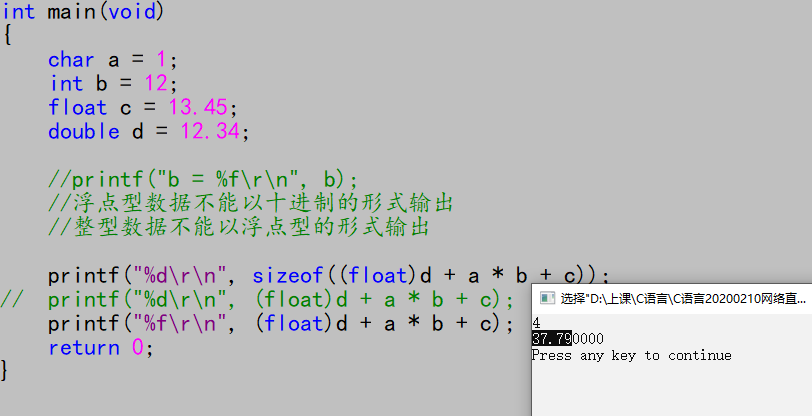
int b = 12;

float c = 3.45;

double d = 12.34;

(float)d + a \* b + c ? flaot

(float)d + (float)(a \* b) + c



7 优先级：只比逗号运算符高

8 结合性：从右到左 a = b = 10;

### 算术运算符

1. 符号： + - \* / %(取余)
2. 表达式：表达式1 算术运算符 表达式2 12 / 3
3. 特殊用法

除法：分母不为0

当除法两边都为整型数据时，结果取整 10 / 3 = 3

当除法有一边为浮点型数据时，结果保留精确值 10.0 / 3 = 3.333 10 / 3.0 10.0 / 3.0

取余：

取余两边必须为整型数据 10 % 3 = 1 10.0/3(不能)

1. 优先级：\* / % 高于 + -
2. 结合性：从左往右
3. 应用
4. 可以分离数据 % /

15：

1：15/10 = 1

5：15%10= 5

45689

4：45689/10000 = 4

取最高位：除以本身位数

取最低位：取余10

12356

1：12356/10000 = 1

2：

放到最高位：12356%10000 = 2356 2356/1000 = 2 12356%10000/1000 = 2

放到最低位：12356/1000 = 12 12%10 = 2 12356/1000%10 = 2

3：

12356%1000/100 = 3

12356/100%10 = 3

5：

6：12356%10 = 6

1. 判断倍数关系 %

10 % 3 != 0 10不是3的倍数 10不能被3整除

12 % 3 == 0 12是3的倍数 12能被3整除

闰年成立的两个条件

* + - 1. 能被4整除，同时不能被100整除
      2. 能被400整除

### 关系运算符

1. 符号：> < >= <= != ==
2. 表达式：表达式1 关系运算符 表达式2 a > b 1 > 2
3. 关系表达式的值
4. 关系成立，关系表达式的值为1
5. 关系不成立，关系表达式的值为0
6. > 2 ? 0
7. < 3 > 2 ? 0

1 < 2 < 3 ? 1

1 < a < 10? 1 可以表示范围？？a = 11 不能 a > 1 && a < 10

1. 优先级：> >= < <= 高于 != ==
2. 结合性：从左往右
3. 作用：只能判断一个条件

### 位运算符

1. 符号：& | ~ ^ << >>
2. 说明：进行位运算前，要把数据转化为二进制 10&20
3. 按位取反 ~
4. 表达式：~(表达式)
5. 规则：1变0，0变1
6. 举例

int a = 0xffffffff;

1111 1111 1111 1111 1111 1111 1111 1111 0xffffffff

~a

0000 0000 0000 0000 0000 0000 0000 0000 0

1. 按位与 &
2. 表达式：表达式1 & 表达式2
3. 规则：全1为1，有0为0
4. 举例

int a = 0xffffffff;

int b = 0xf;

1111 1111 1111 1111 1111 1111 1111 1111 0xffffffff

0000 0000 0000 0000 0000 0000 0000 1111 0xf

a & b

0000 0000 0000 0000 0000 0000 0000 1111 0xf

1. 按位或 |
2. 表达式：表达式1 | 表达式2
3. 规则：有1为1，全0为0
4. 举例

int a = 0xffffffff;

int b = 0xf;

1111 1111 1111 1111 1111 1111 1111 1111 0xffffffff

0000 0000 0000 0000 0000 0000 0000 1111 0xf

a | b

1111 1111 1111 1111 1111 1111 1111 1111 0xffffffff

1. 按位异或 ^
2. 表达式：表达式1 ^ 表达式2
3. 规则：相同为0，不同为1
4. 举例
5. int a = 0xffffffff;

int b = 0xf;

1111 1111 1111 1111 1111 1111 1111 1111 0xffffffff

0000 0000 0000 0000 0000 0000 0000 1111 0xf

a ^ b

1111 1111 1111 1111 1111 1111 1111 0000 0xfffffff0

1. 整体左移 <<
2. 表达式：表达式<<n n为左移的位数
3. 规则：高位溢出，低位补0
4. 举例

int a = 0xffffffff;

1111 1111 1111 1111 1111 1111 1111 1111 0xffffffff

a << 5;

1111 1111 1111 1111 1111 1111 1110 0000 0xffffffe0

1. 整体右移 >>
2. 表达式：表达式>>n n右移的位数
3. 规则：有符号：低位溢出，高位补符号位

无符号：低位溢出，高位补0

1. 举例

int a = 0xffffffff;//有符号 符号位为1

1111 1111 1111 1111 1111 1111 1111 1111 0xffffffff

a >> 5

1111 1111 1111 1111 1111 1111 1111 1111 0xffffffff

int a = 0x7fffffff;//有符号 符号位为0

0111 1111 1111 1111 1111 1111 1111 1111 0xffffffff

a >> 5

0000 0011 1111 1111 1111 1111 1111 1111 0x3ffffff

unsigned int a = 0xffffffff;

1111 1111 1111 1111 1111 1111 1111 1111 0xffffffff

a >> 5

0000 0111 1111 1111 1111 1111 1111 1111 0x7ffffff

1. 优先级：~ 高于 << >> 高于 & 高于 ^ 高于 |
2. 结合性：~从右到左 其它从左到右
3. 位运算的应用
4. 位运算符的操作 X表示0或者1

x & 1 🡪 x

x & 0 🡪 0 清零

x | 1 🡪 1 置一

x | 0 🡪 x

x ^ 0 🡪 x

x ^ 1 🡪 -x 翻转

1. 位运算符配合使用

unsigned char a; 1字节 8bit xxxx xxxx

* 1. 实现xx00 xxxx，前提：不能改变其它位

xxxx xxxx

&

1100 1111 🡨 ~(0011 0000) 🡨 ~(0000 0011 << 4) 🡨 ~(0x03 << 4)

xx00 xxxx

* 1. 实现xx11 xxxx,不能改变其它位

xxxx xxxx

|

0011 0000 🡨 (0000 0011 << 4) 🡨 (0x03 << 4)

xx11 xxxx

* 1. 实现x101 xxxx，不能改变其它位

先清零后赋值

xxxx xxxx

&

1000 1111 🡨 ~(0111 0000) 🡨 ~(0000 0111 << 4) 🡨 ~(0x07 << 4)

x000 xxxx

|

0101 0000 🡨 (0000 0101 << 4) 🡨 (0x05 << 4)

x101 xxxx

* 1. 实现1010 xxxx，不能改变其它位

清零：需要清多少个0，就需要多少个1来取反

//清一个位0 0x01

//清两个位0 0x03

//清三个位0 0x07

//清四个位0 0x0f

//清八个位0 0xff

xxxx xxxx a

a = a & ~(0x0f << 4); 🡪 a &= ~(0x0f << 4); a += b; == a = a + b;

0000 xxxx

a = a | (0x0a << 4); 🡪 a |= (0xa << 4);

### 逻辑运算符

1. 符号： && || ！

2. 逻辑值：数据是非零值，逻辑值是真；数据是零值，逻辑值就假

3. 逻辑表达式的值：逻辑值为真，逻辑表达式的值为1

逻辑值为假，逻辑表达式的值为0

4 逻辑与 &&

(1). 逻辑与表达式：表达式1 && 表达式2 a > 1 && a < 10

(2). 规则：全真为真，一假则假

(3). 原理：当表达式1的逻辑值为假，则程序不会执行表达式2，那么逻辑与表达式的值为0

当表达式1的逻辑值为真，程序会去执行表达式2

如果表达式2的逻辑值为真，那么逻辑与表达式的值为1

如果表达式2的逻辑值为假，那么逻辑与表达式的值为0

1. 举例

int a = 1;

int b = 2;

int c = 5;

int d = 0;

d = (a > b) && (c = 3) 优先级和结合性

d = 0 c = 5

d = (a < b) && (c = 3)

d = 1 c = 3

1. 逻辑或 ||

(1). 逻辑或表达式：表达式1 || 表达式2 a > 1 || a < 10 a = 0

(2). 规则：有真为真，全假则假

(3). 原理：当表达式1的逻辑值为真，则程序不会执行表达式2，那么逻辑或表达式的值为1

当表达式1的逻辑值为假，程序会去执行表达式2

如果表达式2的逻辑值为真，那么逻辑或表达式的值为1

如果表达式2的逻辑值为假，那么逻辑或表达式的值为0

1. 举例

int a = 1;

int b = 2;

int c = 5;

int d = 0;

d = (a > b) || (c = 3)

d = 1 c = 3

d = (a < b) || (c = 3)

d = 1 c = 5

1. 逻辑非 ！(单目运算符)
2. 逻辑非表达式：!(表达式) !(a > 5) = 1 a = 2;
3. 规则：真变假，假变真
4. 举例

int a = 1;

int b = 2;

int c = 5;

int d = 0;

d = !(a > b) || (c = 3)

d = 1 c = 5

d = !(a < b) || (c = 3)

d = 1 c = 3

1. 优先级：! 高于 && 高于 ||
2. 结合性：!(从右往左) &&、||从左往右

9 作用：将关系表达式连接起来，可以判断多个条件

课堂练习

* + - 1. 判断闰年

能被4整除的同时不能被100整除 &&

能被400整除 ||

* + - 1. 输入5位数，判断该数是不是回文数(正读和反读一样) 12321 12345 23532

### 自增和自减运算符

1. 符号：++ --
2. 说明：前置自增自减、后置自增自减
3. 作用：单位加一，单位减一
4. 使用
   * + - 1. 不管是前置还是后置，都是单位加一减一
         2. 前置：先自增，再使用 b = ++a; a = a + 1; b = a;
         3. 后置：先使用，再自增(遇到分号) b = a++ b = a a = a + 1
5. 注意
6. 自增自减只能是变量
7. 通常放在循环体内
8. 当一个表达式里面出现多个自增或者自减，程序员不会这样写，但是面试有可能会考

### 复合运算符

1. 符号：+= -= /= \*=
2. 表达式：变量 复合运算符 表达式

a = a + 3; a += 3

a = a \* 4； a \*= 4

a \*= 1 + 4； a = a \* (1 + 4)

1. 优先级：跟赋值运算符
2. 结合性：从右到左
3. 说明：是不是所以的运算符都可以跟赋值运算符组合成复合运算符？

本身带有赋值运算的就不行 == != >= ….

### 条件运算符

1. 符号：?:
2. 表达式：(关系表达式)?(表达式1):(表达式2)
3. 原理：如果关系表达式成立，则表达式1就为条件表达式的值

如果关系表达式不成立，则表达式2就为条件表达式的值

1. 优先级：高于赋值运算符、条件运算符
2. 结合性：从右到左

### 逗号运算符

1. 符号：，
2. 表达式：变量 = (表达式1，表达式2….表达式n);

变量 = 表达式1；

变量 = 表达式2；

变量 = 表达式n；

1. 原理：最终给变量的是最后一个表达式的值
2. 注意：前面表达式有参与计算
3. 优先级：最低
4. 结合性：从左到右

总结：

单目运算符 > 算术运算符 > 关系运算符 > 按位运算符 > 逻辑运算符 > 条件运算符 > 赋值运算符