



知识点总结



·常用的运算符有：

- 1、**算术运算符**（++、--、+、-、*、/、%）：用于进行基本的算术运算。
- 2、**关系运算符**（<、<=、>、>=、==、!=）：用于比较两个值或表达式，返回真或假。
- 3、**逻辑运算符**（!、&&、||）：常用于条件判断，返回真或假。
- 4、**位运算符**（~、<<、>>、&、^、|）：用于对整数类型的数据进行按位操作。
- 5、**赋值运算符**（=、+=、-=、*=、/=、%=、<<=、>>=、&=、^=、|=）：用于构成赋值表达式。
- 6、**条件运算符**（?:）：用于构成三元表达式，根据条件的真假选择不同的值，常用于简化if-else语句。
- 7、**逗号运算符**（,）：用于将多个表达式按从左到右的顺序执行，并返回最后一个表达式的值。



6、**条件运算符** (`?:`)：用于构成三元表达式，根据条件的真假选择不同的值，常用于简化if-else语句。

语法格式：**条件表达式 ? 表达式1 : 表达式2;**

当条件为真时，该三元表达式的值为?后面的表达式1的值。

当条件为假时，该三元表达式的值为:后面的表达式2的值。

例子：`int max = (a > b) ? a : b;`

当 `a > b` 为真时，该三元表达式的值为a，max被赋值为a。

当 `a > b` 为假时，该三元表达式的值为b，max被赋值为b。

7、**逗号运算符** (,)：用于将多个表达式按从左到右的顺序执行，并**返回最后一个表达式的值**。语法格式：**表达式1, 表达式2, 表达式3, ..., 表达式n**



例如：

```
#include <stdio.h>
int main() {
    int a = 2;
    int b = 3;
    // 使用逗号运算符将多个表达式组合
    int c = (a++, b++, a + b);
    printf("c 的值为： %d\n", c);
}
```

程序执行到 `int c = (a++, b++, a + b);` 这行代码时：

会先计算 `a++`，`a` 变为 3；

接着计算 `b++`，`b` 变为 4；

最后计算 `a + b`，即 $3 + 4 = 7$ 。

而逗号运算符返回的是最后一个表达式的值，所以整个逗号表达式的值为 7，因此 `c` 被赋值为 7。



·按照操作数（**具体值**、**变量**、**表达式**等）的个数，运算符可以分为三类：

- 1、**单目运算符**：只对一个操作数进行操作。比如，**!**、**~**、**++**、**--**等。
- 2、**双目运算符**：对两个操作数进行操作。比如，**+**、**-**、*****、**/**、**<**、**<=**等。
- 3、**三目运算符**：对三个操作数进行操作。比如，条件运算符（**? :**）。

在对表达式进行运算时，要遵循运算符的**优先级**和**结合性**规则，以确定运算顺序。

优先级：当运算符的优先级不同时，优先级高的运算符会先进行运算。

结合性：当运算符的优先级相同时，结合性就决定了运算的方向。

结合性分为**左结合性（从左到右计算）**和**右结合性（从右到左计算）**。

比如，加减运算符具有左结合性，对于表达式 $a - b + c$ ，会先计算 $a - b$ 的结果，再将就这个结果与 c 相加。

赋值运算符具有右结合性，对于表达式 $a = b = c$ ，会先计算 $b = c$ ，再计算 $a = b$ 。



·这些常用运算符的优先级（从高到低）以及结合性情况如下表所示：

优先级
↓

类别	运算符	结合性
圆括号	()	从左到右
单目运算符	+（正号）、-（负号）、！（逻辑非）、~（按位取反）、++（自增）、--（自减）	从右到左
乘除	*（乘法）、/（除法）、%（取余）	从左到右
加减	+（加法）、-（减法）	从左到右
移位	<<（左移）、>>（右移）	从左到右
大小关系	<（小于）、<=（小于等于）、>（大于）、>=（大于等于）	从左到右
相等	（等于）、!=（不等于）	从左到右
按位与	&	从左到右
按位异或	^	从左到右
按位或		从左到右
逻辑与	&&	从左到右
逻辑或		从左到右
条件运算符	? :	从右到左
赋值	=(赋值)、+=(加法赋值)、-=(减法赋值)、*=(乘法赋值)、/=(除法赋值)、	从右到左
逗号	,	从左到右



圆括号的优先级最高，接着是自增运算符，然后是逻辑与运算符，最后是条件运算符。
逻辑与具有短路特性，左边表达式为假时，整个逻辑表达式的值必定为假。



题目1、若有定义：`int a = 0, b = 0, c = 0, d = 0;`

有C语言表达式 `(a++ && b++) ? c++ : d++`，以下关于其执行顺序的叙述正确是（ A ）

- A. 先执行`a++`，表达式`a++`的值为0，由此即可确定`(a++ && b++)`值为0，因此执行`d++`
- B. 先执行`a++`，表达式`a++`的值为0；再执行`b++`，表达式`b++`的值为0，由此可确定`(a++ && b++)`值为0，因此执行`d++`
- C. 先执行`a++`，表达式`a++`的值为1；再执行`b++`，表达式`b++`的值为1，由此可确定`(a++ && b++)`值为1，因此执行`c++`
- D. 先执行`b++`，表达式`b++`的值为1；再执行`a++`，表达式`a++`的值为1，由此可确定`(a++ && b++)`值为1，因此执行`c++`



圆括号的优先级最高

关系运算符的优先级高于逻辑或运算符



题目2、下列关系表达式中，结果为“假”的是（ **B** ）

A. $(3 + 4) > 6$

B. $(3 \neq 4) > 2$

C. $3 \leq 4 \parallel 3$

D. $(3 < 4) == 1$



算术运算符优先级高于赋值运算符



题目3、有以下程序：

```
#include <stdio.h>
```

```
main()
```

```
{
```

```
    int A = 0, B = 0, C = 0;
```

```
    C = (A -= A - 5);
```

```
    (A = B, B += 4);
```

```
    printf("%d,%d,%d\n", A, B, C);
```

```
}
```

程序运行后输出的结果是 (A)

A. 0,4,5

B. 4,4,5

C. 4,4,4

D. 0,0,0



条件运算符是右结合的，当有多个条件运算符在一起，它们会从右边开始结合。



题目4、以下程序的运行结果是（ D ）。

```
main()  
{  
    int k = 4, a = 3, b = 2, c = 1;  
    printf("\n%d\n", k < a ? k : c < b ? c : a);  
}
```

A. 4

B. 3

C. 2

D. 1



题目5、以下程序段中，与语句： $k = a > b ? (b > c ? 1 : 0) : 0;$ 功能相同的是 (A)

A.

```
if((a > b) && (b > c)) k = 1;  
else k = 0;
```

B.

```
if ((a > b) || (b > c)) k = 1;  
else k = 0;
```

$a > b$ 或 $b > c$, k 赋值为1

C.

```
if (a <= b) k = 0;  
else if (b <= c) k = 1;
```

$a \leq b$ 时, k 赋值为0, $a > b$ 且 $b \leq c$ 时, k 赋值为1

D.

```
if (a > b) k = 1;  
else if (b > c) k = 1;  
else k = 0;
```

$a > b$ 时, k 赋值为1, $a \leq b$ 且 $b > c$ 时, k 赋值为1