一、基本运算

【大纲内容】

【难度 1】算术运算：加、减、乘、除、整除、求余

【难度 1】关系运算：大于、大于等于、小于、小于等于、等于、不等于

【难度 1】逻辑运算：与(&&)、或(II)、非(!)

【难度 1】变量自增与自减运算

【难度 1】三目运算

【难度 2】位运算：与(&)、或(1)、非(~)、异或(^)、左移(<<)、右移(>>)

✨ 运算符（Operator）说明

在 C++ 中，“运算符”就像数学里的符号（+ - × ÷ =），可以用来计算数字、判断对错、组合条件。

（一）算术运算（Arithmetic Operators）

☑ 用来进行数学计算

运算符	中文名	功能说明	举例	举例子
+	加法	把两个数加起来	3 + 4 = 7	拿 3 个苹果，再加 4 个苹果
-	减法	两个数相减	10 - 6 = 4	有 10 元，花掉 6 元，还剩 4 元
*	乘法	两个数相乘	2 * 5 = 10	每袋 5 颗糖，2 袋共 10 颗
/	除法	一个数除以另一个	9 / 2 = 4	9 颗糖每人分 2 颗，还能分几人?
%	求余（取余）	除法后剩下的余数	9 % 2 = 1	9 除以 2，每人 2 颗，剩 1 颗

☑ 示例代码：

```
int a = 9, b = 4;
cout << a + b << endl; // 13
cout << a - b << endl; // 5
cout << a * b << endl; // 36
cout << a / b << endl; // 2（整数除法，不保留小数）
cout << a % b << endl; // 1（9 ÷ 4 余 1）
```

考试注意事项 & 易错点：

❗整数除法 / 结果只保留整数部分，不会自动变小数！

```
cout << 7 / 2 << endl; // 输出 3，不是 3.5！
```

❗a % b 中 b 不能是 0，不能除以 0，否则出错！

（二）关系运算（Relational Operators）

☑ **用来比较两个值，结果是 true 或 false**

运算符	中文名	英文术语	示例（a=5, b=3）	结果
>	大于	greater than	a > b	true
<	小于	less than	b < a	true
>=	大于等于	greater or equal	a >= b	true
<=	小于等于	less or equal	b <= a	true
==	等于	equal to	a == b	FALSE
!=	不等于	not equal to	a != b	true

☑ **示例代码：**

```
int a = 5, b = 3;
if (a > b) {
```

```
        cout << "a 比 b 大" << endl;
    }
```

通俗解释：

- > 就像：“你比我高吗？”
- == 就像：“你和我一样高吗？”
- != 就是：“你和我不一样吧？”

考试注意事项 & 易错点：

错误写法	正确写法	问题说明
=	==	= 是赋值，== 是判断相等
忘写小于号等	<=	小于等于必须连写 <=，不能写 < =

（三）逻辑运算（Logical Operators）

☑ **用来连接多个条件**

运算符	中文名	英文名	功能说明
&&	与	and	两个条件都为真，结果才是真
||	或	or	两个条件都为假，结果才是假
!	非	not	条件取反，原来是 true → false，false → true

☑ **示例代码：**

```
int age = 10;
if (age >= 6 && age <= 12) {
    cout << "你是小学生！" << endl;
}
if (age < 6 || age > 12) {
```

```
    cout << "你不是小学生。" << endl;
}
bool isSunny = false;
if (!isSunny) {
    cout << "今天天气不好。" << endl;
}
```

考试注意事项 & 易错点：

- || 是两个竖线（Enter 键 1 上边，按 Shift）
- && 是两个&号，不是一个 &
- ！是感叹号，不是字母 l、小写 L、或大写 i
- 逻辑运算只能用在**判断**中，不能用来算数！

综合练习题示例：

```
int age = 8;
bool isSunny = true;
if (age >= 6 && age <= 12 && isSunny) {
    cout << "你可以在阳光下玩耍！" << endl;
}
```

总结表格

类型	符号	功能说明	示例
加法	+	相加	3 + 2 = 5
减法	-	相减	5 - 1 = 4
乘法	*	相乘	2 * 3 = 6
除法	/	相除（只保留整数）	7 / 2 = 3
余数	%	除法后的“剩下的”	7 % 2 = 1

等于	==	判断是否一样	a == b
不等于	!=	判断是否不一样	a != b
大于	>	左边是否比右边大	a > b
小于等于	<=	左边是否小于或等于右边	a <= b
与	&&	两个都对才是真的	a > 0 && a < 10
或	||	两个都假才是假的	a > 0 || a < 10
非	!	取反	!true = false

（四）变量自增与自减运算（Increment / Decrement）

- 自增（Increment）：变量的值自动+1，使用 ++
- 自减（Decrement）：变量的值自动−1，使用 --

两种写法：

写法	名称	示例	说明
i++	后缀自增	int i = 1;i++;	先用 i，再让 i 加 1
++i	前缀自增	int i = 1;++i;	先让 i 加 1，再使用 i 的值
i--	后缀自减	int i = 5;i--;	先用 i，再让 i 减 1
--i	前缀自减	int i = 5; --i;	先减 1，再使用 i 的值

- i++ 就像吃完一颗糖再从糖罐子里拿一颗；
- ++i 是先从糖罐子里拿一颗再吃。

☑ 示例代码：

```
int x = 5;
cout << x++ << endl; // 输出 5，然后 x 变成 6
```

```
cout << ++x << endl; // x 先变成 7，然后输出 7
```

考试易错点：

- i++ 和 ++i 看起来差不多，但顺序不同！
- 不要写成 i+ +（中间多了空格，会出错）
- 只能对变量使用 ++ 和 --，不能对常量！

```
int a = 3;
3++; // ❌ 错误，不能对 3 这种数直接加加
```

（五）三目运算（Ternary Operator）

三目运算符是 **条件 ？ 表达式 1 ： 表达式 2**，根据条件判断选择结果。

语法结构：

```
结果 = 条件 ？ 如果为真时的值 ： 如果为假时的值;
```

比喻：

- 妈妈说：“如果你考了 100 分，就吃冰淇淋；否则吃胡萝卜。”
- 这就可以写成三目运算。

☑ 示例代码：

```
int score = 100;
string food = (score == 100) ? "冰淇淋" : "胡萝卜";
cout << "你吃：" << food << endl;
```

- 说明：如果 score == 100 为真，就吃冰淇淋；否则吃胡萝卜。

考试易错点：

- 三目运算必须有两个选项（真和假），不能漏掉 ：
- 三目运算结果可以赋值给变量，也可以直接输出：

```
cout << (a > b ? a : b); // 输出较大的数
```

（六）位运算（Bitwise Operators）

位运算是对**整数的二进制**表示进行操作的方式。虽然看起来高级，但其实可以像拼图一样理解。

想象数字是由 0 和 1 拼成的“灯泡串”，1 是亮灯，0 是灭灯。位运算就像控制这些灯的开关！

☑常见位运算及含义：

运算符	中文名	说明	示例 (5 = 0101, 3 = 0011)
&	位与	两个灯都亮才亮(1&1=1,其它是 0)	5 & 3 = 1 (0001)
|	位或	只要有一个灯亮就亮（1 或 1=1）	5 | 3 = 7 (0111)
~	位非	灯亮就灭，灯灭就亮（取反）	~5 = -6 (按补码处理)
^	异或	不一样才亮（1^0=1，1^1=0）	5 ^ 3 = 6 (0110)
<<	左移	全体灯往左挪（相当于 ×2）	5 << 1 = 10 (1010)
>>	右移	全体灯往右挪（相当于 ÷2）	5 >> 1 = 2 (0010)

☑示例代码：

```
int a = 5, b = 3;
cout << (a & b) << endl; // 1
cout << (a | b) << endl; // 7
```

```
cout << (a ^ b) << endl; // 6
cout << (a << 1) << endl; // 10
cout << (a >> 1) << endl; // 2
```

考试注意事项 & 易错点：

易错点	正确写法	说明
位或输入有误 I	用 Shift+\	\键在 Enter 键上面
~ 不是减号	~ 是波浪号	位非不是 -，在 1 键左边
位运算只能用于整数	int 类型	不能对浮点数、字符串进行位运算
<< >> 是成对写	不是单个 <	少写一个符号会变成比较运算了