

03 - 表达式与运算

C++ 程序设计基础

SOJ 信息学竞赛教练组

2024 年 6 月 6 日

目录

1 复习回顾

2 算术运算

3 表达式

4 赋值运算

5 关系运算

6 逻辑运算

7 总结

变量的类型

- 整数类型
 - int: 近似值 2.1×10^9
 - long long: 近似值 9.2×10^{18}

变量的类型

- 整数类型
 - int: 近似值 2.1×10^9
 - long long: 近似值 9.2×10^{18}
- 浮点数类型（实数/小数）
 - double: 双精度浮点数

变量的类型

- 整数类型
 - int: 近似值 2.1×10^9
 - long long: 近似值 9.2×10^{18}
- 浮点数类型（实数/小数）
 - double: 双精度浮点数
- 字符类型
 - char: 键盘上的字母、数字及符号

变量的类型

- 整数类型
 - int: 近似值 2.1×10^9
 - long long: 近似值 9.2×10^{18}
- 浮点数类型（实数/小数）
 - double: 双精度浮点数
- 字符类型
 - char: 键盘上的字母、数字及符号
- 布尔类型
 - bool: true / false（真 / 假）

变量的声明与访问

- 变量的声明
 - 类型 变量名
 - `int a;`

变量的声明与访问

- 变量的声明
 - 类型 变量名
 - `int a;`
- 变量的赋值
 - `a = 100;`

变量的声明与访问

- 变量的声明
 - 类型 变量名
 - `int a;`
- 变量的赋值
 - `a = 100;`
- 使用 `cin` 语句输入变量，在输入流符号 `>>` 后写上变量名
 - `cin >> a;`

变量的声明与访问

- 变量的声明
 - 类型 变量名
 - `int a;`
- 变量的赋值
 - `a = 100;`
- 使用 `cin` 语句输入变量，在输入流符号 `>>` 后写上变量名
 - `cin >> a;`
- 使用 `cout` 语句输出变量，在输出流符号 `<<` 后写上变量名
 - `cout << a << endl;`

目录

1 复习回顾

2 算术运算

3 表达式

4 赋值运算

5 关系运算

6 逻辑运算

7 总结

算术运算

操作	操作符	例子	结果
加法	+	$20 + 10$	30
减法	-	$20 - 10$	10
乘法	*	$20 * 10$	200
除法	/	$20 / 10$	2
取模 (求余数)	%	$20 \% 10$	0

- **相同数据类型的数据进行算术运算，得到的结果也是相同类型**

- **相同数据类型的数据进行算术运算，得到的结果也是相同类型**
 - 例：整数除以整数，结果也是整数

- **相同数据类型的数据进行算术运算，得到的结果也是相同类型**
 - 例：整数除以整数，结果也是整数
 - $2 / 4$ 的结果是什么？是 0 还是 0.5？

- **相同数据类型的数据进行算术运算，得到的结果也是相同类型**
 - 例：整数除以整数，结果也是整数
 - $2 / 4$ 的结果是什么？是 0 还是 0.5？
 - 是 0（整数）而不是 0.5（实数）

- **相同数据类型的数据进行算术运算，得到的结果也是相同类型**
 - 例：整数除以整数，结果也是整数
 - $2 / 4$ 的结果是什么？是 0 还是 0.5？
 - 是 0（整数）而不是 0.5（实数）
 - 那如何让 $2 / 4$ 的结果为 0.5 呢？

隐式类型转换

- 不同数据类型的数据进行算术运算，则存储空间小的类型自动转换为存储空间大的类型，再进行运算，运算的结果为存储空间大的类型

隐式类型转换

- **不同数据类型**的数据进行算术运算，则存储空间小的类型自动转换为存储空间大的类型，再进行运算，运算的结果为**存储空间大的类型**
 - 例：实数除以整数，结果是实数

隐式类型转换

- 不同数据类型的数据进行算术运算，则存储空间小的类型自动转换为存储空间大的类型，再进行运算，运算的结果为存储空间大的类型
 - 例：实数除以整数，结果是实数
 - $2.0 / 4$ 的结果的结果为 0.5（实数）

选择题

1. 下面语句中可以得到整数的有

- A. `10 / 4.0;`
- B. `(double)10 / 4;`
- C. `1.0 * 10;`
- D. `10 * 4;`

选择题

1. 下面语句中可以得到整数的有

- A. `10 / 4.0;`
- B. `(double)10 / 4;`
- C. `1.0 * 10;`
- D. `10 * 4;`

目录

1 复习回顾

2 算术运算

3 表达式

4 赋值运算

5 关系运算

6 逻辑运算

7 总结

表达式

- 表达式，是由变量、常量、运算符和分组符号等组成，能通过计算得出数值的式子，如 $a * 10 - b$

表达式

- 表达式，是由变量、常量、运算符和分组符号等组成，能通过计算得出数值的式子，如 $a * 10 - b$
 - 分组符号只可以为小括号，如 $a * (10 - b)$

- 表达式，是由变量、常量、运算符和分组符号等组成，能通过计算得出数值的式子，如 $a * 10 - b$
 - 分组符号只可以为小括号，如 $a * (10 - b)$
 - 运算符不可以省略不写

- 表达式，是由变量、常量、运算符和分组符号等组成，能通过计算得出数值的式子，如 $a * 10 - b$
 - 分组符号只可以为小括号，如 $a * (10 - b)$
 - 运算符不可以省略不写
 - 括号使用时，需要成对出现

选择题

1. 以下选项中出现的字母或单词均为变量或常量，那么以下选项中在 C++ 中属于合法的表达式的有

A. $2(a + b)$

B. $[(a + b) * h] / 2$

C. $2pi * r$

D. $pi * r * r$

选择题

1. 以下选项中出现的字母或单词均为变量或常量，那么以下选项中在 C++ 中属于合法的表达式的有

A. $2(a + b)$

B. $[(a + b) * h] / 2$

C. $2\pi * r$

D. $\pi * r * r$

表达式

- 在表达式中，参与运算不会改变变量的值
- 对于表达式的结果，我们可以直接输出，或存储后再使用

表达式

- 在表达式中，参与运算不会改变变量的值
- 对于表达式的结果，我们可以直接输出，或存储后再使用

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int a = 5;
7     a / (3.0 - 1);    // 无意义
8     cout << a << endl; // 输出 5
9
10    return 0;
11 }
```

表达式

- 在表达式中，参与运算不会改变变量的值
- 对于表达式的结果，我们可以直接输出，或存储后再使用

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int a = 5;
7     cout << a / (3.0 - 1) << endl; // 输出 2.5
8     cout << a << endl;           // 输出 5
9
10    return 0;
11 }
```


运算的优先级

- 在表达式求值时，必须了解各种运算及其优先顺序
 - 优先顺序从高到低排列：
 1. 括号
 2. 乘法 / 除法 / 取模
 3. 加法 / 减法
 4. 关系运算
 5. 逻辑运算
 6. 赋值运算
 - 同等优先顺序的两个运算则按照从左到右进行计算
 - 除了赋值运算和逻辑非

目录

1 复习回顾

2 算术运算

3 表达式

4 赋值运算

5 关系运算

6 逻辑运算

7 总结

赋值运算

操作	操作符	解释
赋值	=	将结果赋予变量
加法赋值	+=	$a += b$, 等价于 $a = a + b$
减法赋值	-=	$a -= b$, 等价于 $a = a - b$
乘法赋值	*=	$a *= b$, 等价于 $a = a * b$
除法赋值	/=	$a /= b$, 等价于 $a = a / b$
取模赋值	%=	$a \% = b$, 等价于 $a = a \% b$

填空题

1. 在各表达式互相独立、不相互影响的情况下，写出执行下列表达式后各变量的值。(a、b 的初始值分别为 21、5)
 - $a + b$;
 - $a += b$;
 - $b = a / b$;
 - $a \% = b$;

填空题

1. 在各表达式互相独立、不相互影响的情况下，写出执行下列表达式后各变量的值。(a、b 的初始值分别为 21、5)

- $a + b;$ $a = 21$ $b = 5$
- $a += b;$ $a = 26$ $b = 5$
- $b = a / b;$ $a = 21$ $b = 4$
- $a \% = b;$ $a = 1$ $b = 5$

例 3.1：交换两个变量的值

编程题

- 输入两个整数 a, b ($1 \leq a, b \leq 10^3$), 输出交换后的整数。
- 样例输入
3 5
- 样例输出
5 3

例 3.1：交换两个变量的值

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int a, b;
7     cin >> a >> b;
8     int tmp = a; // 把变量 a 原本的值存到变量 tmp 中
9     a = b;       // 把变量 b 的值赋值给变量 a
10    b = tmp;      // 把变量 tmp 的值赋值给变量 b
11    cout << a << " " << b << endl;
12
13    return 0;
14 }
```

自增自减

操作	操作符	解释
自增	++	变量的值增加 1
自减	--	变量的值减少 1

自增自减

操作	操作符	解释
自增	++	变量的值增加 1
自减	--	变量的值减少 1

- 自增和自减运算符可放在变量前或变量后，单独使用时，效果都一样
 - $a++$; 与 $++a$; 等价，都表示 $a = a + 1$;
 - $a--$; 与 $--a$; 等价，都表示 $a = a - 1$;

- 自增和自减运算符参与其他操作时，运算符的前后位置会影响结果

- 自增和自减运算符参与其他操作时，运算符的前后位置会影响结果
 - `b = a++;`
 - 先赋值，再 `a++`
 - 相当于 `b = a; a = a + 1;`

- 自增和自减运算符参与其他操作时，运算符的前后位置会影响结果
 - `b = a++;`
 - 先赋值，再 `a++`
 - 相当于 `b = a; a = a + 1;`
 - `b = ++a;`
 - 先 `++a`，再赋值
 - 相当于 `a = a + 1; b = a;`

字符的简单运算

- 字符类型的变量，也可以进行加减运算
 - `char ch = 'a';` // 字符类型变量 `ch` 赋值为字母 `a`
 - `ch++;` // 变量 `ch` 的值加 1
 - `cout << ch << endl;` // 输出字母 `b`

例 3.2：输出下一个字母

编程题

- 输入一个 a 到 y 之间的小写字母，输出该字母的下一个字母。
- 样例输入
a
- 样例输出
b

例 3.2：输出下一个字母

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     char ch;
7     cin >> ch;
8     ch++;
9     cout << ch << endl;
10
11     return 0;
12 }
```

目录

1 复习回顾

2 算术运算

3 表达式

4 赋值运算

5 关系运算

6 逻辑运算

7 总结

关系运算

操作	操作符	例子	结果
判断相等	<code>==</code>	<code>20 == 10</code>	<code>false</code>
判断不等	<code>!=</code>	<code>20 != 10</code>	<code>true</code>
判断大于	<code>></code>	<code>20 > 10</code>	<code>true</code>
判断小于	<code><</code>	<code>20 < 10</code>	<code>false</code>
判断大于等于	<code>>=</code>	<code>20 >= 10</code>	<code>true</code>
判断小于等于	<code><=</code>	<code>20 <= 20</code>	<code>true</code>

- 参与关系运算的元素是任意类型的表达式，结果是 bool 类型

关系运算

- 参与关系运算的元素是任意类型的表达式，结果是 bool 类型
- 可以将关系运算表达式的结果赋值给 bool 类型变量并输出

关系运算

- 参与关系运算的元素是任意类型的表达式，结果是 bool 类型
- 可以将关系运算表达式的结果赋值给 bool 类型变量并输出
 - `bool f = (20 > 10);`
`cout << f << endl;` // 输出结果为 1

关系运算

- 参与关系运算的元素是任意类型的表达式，结果是 bool 类型
- 可以将关系运算表达式的结果赋值给 bool 类型变量并输出
 - `bool f = (20 > 10);`
`cout << f << endl;` // 输出结果为 1
 - `bool f = (10 + 10 != 20);`
`cout << f << endl;` // 输出结果为 0

判断倍数关系

- 如何判断整数 a 是否为 2 的倍数?

判断倍数关系

- 如何判断整数 a 是否为 2 的倍数?
 - `bool f = (a % 2 == 0);`
`cout << f << endl;` // 2 的倍数则输出 1, 否则输出 0

例 3.3: 4 的倍数

编程题

- 输入一个整数 n ($0 \leq n \leq 10^9$), 如果该整数是 4 的倍数, 则输出 1, 否则输出 0。
- 样例输入
12
- 样例输出
1

例 3.3: 4 的倍数

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int n;
7     cin >> n;
8     bool f = (n % 4 == 0);
9     // 判断 n 是否为 4 的倍数, 即 n 除以 4 的余数是否为 0
10    cout << f << endl;
11
12    return 0;
13 }
```

目录

1 复习回顾

2 算术运算

3 表达式

4 赋值运算

5 关系运算

6 逻辑运算

7 总结

- 参与逻辑运算的操作数是 bool 类型，结果也是 bool 类型
- 逻辑运算有以下三种：
 - 逻辑与 &&
 - 逻辑或 ||
 - 逻辑非 !

逻辑与 &&

- 逻辑与运算符有“并且”的含义，当两侧的操作数都成立 (true) 时，表达式才成立 (true)

表达式	操作数运算	结果
<code>2 > 1 && 2 < 3</code>	<code>true && true</code>	<code>true</code>
<code>2 > 1 && 2 > 3</code>	<code>true && false</code>	<code>false</code>
<code>2 < 1 && 2 < 3</code>	<code>false && true</code>	<code>false</code>
<code>2 < 1 && 2 > 3</code>	<code>false && false</code>	<code>false</code>

逻辑或 ||

- 逻辑或运算符有“或者”的含义，当两侧的操作数有一个成立 (true) 时，表达式就成立 (true)

表达式	操作数运算	结果
<code>2 > 1 2 < 3</code>	<code>true true</code>	<code>true</code>
<code>2 > 1 2 > 3</code>	<code>true false</code>	<code>true</code>
<code>2 < 1 2 < 3</code>	<code>false true</code>	<code>true</code>
<code>2 < 1 2 > 3</code>	<code>false false</code>	<code>false</code>

逻辑非 !

- 逻辑非运算符会将表达式的结果取反

表达式	操作数运算	结果
<code>!(2 > 1)</code>	<code>!(true)</code>	<code>false</code>
<code>!(2 < 1)</code>	<code>!(false)</code>	<code>true</code>
<code>!(!(2 > 1))</code>	<code>!(!(true))</code>	<code>true</code>

选择题

1. 在 C++ 中, 判断 a 等于 0 或 b 等于 0 的表达式是

A. `a != 0 && b != 0`

B. `a != 0 || b != 0`

C. `a == 0 || b == 0`

D. `a = 0 || b = 0`

选择题

1. 在 C++ 中, 判断 a 等于 0 或 b 等于 0 的表达式是

A. `a != 0 && b != 0`

B. `a != 0 || b != 0`

C. `a == 0 || b == 0`

D. `a = 0 || b = 0`

例 3.4：两位数判断

编程题

- 输入一个整数 n ($0 \leq n \leq 10^9$)，如果该整数是两位数，则输出 1，否则输出 0。
- 样例输入
10
- 样例输出
1

例 3.4：两位数判断

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int n;
7     cin >> n;
8     bool f = (10 <= n && n <= 99);
9     // 两位数是 10 ~ 99 之间的数
10    // 注意不能写成 10 <= n <= 99 的形式
11    cout << f << endl;
12
13    return 0;
14 }
```

目录

1 复习回顾

2 算术运算

3 表达式

4 赋值运算

5 关系运算

6 逻辑运算

7 总结

总结

- 算术运算
- 赋值运算
- 关系运算
- 逻辑运算
- 运算优先级

Thank you!