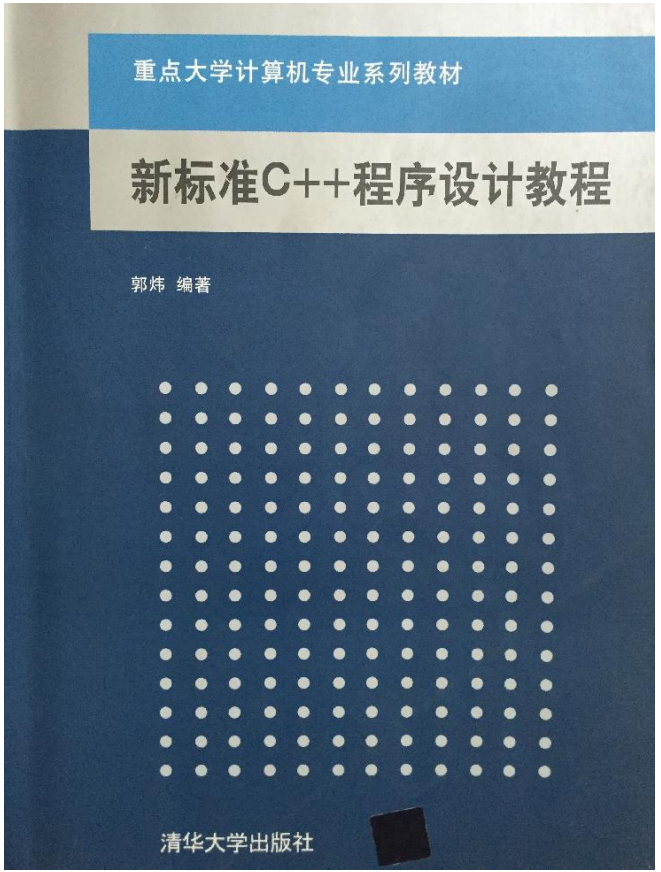
程序设计与算法(一)

李文新 郭炜

主讲教师互动微博：
http://weibo.com/guoweiofpku

指定教材:



赋值运算符、算术运算符和算术表达式

赋值运算符

- 赋值运算符用于给变量赋值，常用有以下六种

=

+=

-=

***=**

/=

%=

赋值运算符

```
int a;
a = 1;          // a的值变为1
a = a + 1;      // a的值变为2
a = 4 + a;      // a的值变为6
a += b ;        //等效于 a = a + b，但是执行速度更快
```

-=, *= ,/= ,%= 用法与+=类似

表达式 x = y 的值，就是y的值

算术运算符

七种算术运算符用于数值运算
运算符+操作数构成表达式

加 +
减 -
乘 *
除 /
求余数 %
自增 ++
自减 --

加、减、乘运算符

a+b、a-b、a*b这三个表达式的值，就是a和b做算术运算的结果。表达式的值的类型，以操作数中精度高的类型为准。

加、减、乘运算符

a+b、a-b、a*b这三个表达式的值，就是a和b做算术运算的结果。表达式的值的类型，以操作数中精度高的类型为准。

精度：

double > long long > int > short > char

加、减、乘运算符

a+b、a-b、a*b这三个表达式的值，就是a和b做算术运算的结果。表达式的值的类型，以操作数中精度高的类型为准。

精度：

double > long long > int > short > char

2 * 0.5 => 1.0

加、减、乘运算的溢出

两个整数类型进行加、减、乘都可能导致计算结果超出了结果类型所能表示的范围，这种情况就叫做溢出。

加、减、乘运算的溢出

两个整数类型进行加、减、乘都可能导致计算结果超出了结果类型所能表示的范围，这种情况就叫做溢出。

计算结果的溢出部分直接被丢弃。

加、减、乘运算的溢出

两个整数类型进行加、减、乘都可能导致计算结果超出了结果类型所能表示的范围，这种情况就叫做溢出。

计算结果的溢出部分直接被丢弃。

实数（浮点数）运算也可能溢出，结果不易预测。

加、减、乘运算的溢出

```
#include <iostream>
using namespace std;
int main()
{
        unsigned int n1 = 0xffffffff;
        cout << n1 << endl;    //输出4294967295
        unsigned int n2 = n1 + 3; //导致溢出
        cout << n2 << endl;    //输出2
        return 0;
} // 0xffffffff + 3 的结果，应该是 0x100000002，
```

加、减、乘运算的溢出

- 有时计算的最终结果似乎不会溢出，但中间结果可能溢出，这也会导致程序出错

加、减、乘运算的溢出

●有时计算的最终结果似乎不会溢出，但中间结果可能溢出，这也会导致程序出错

●例：(a+b)/2 未必等于 a/2+b/2

```
printf("%d",(2147483646 + 6)/2);   => -1073741822
printf("%d",2147483646/2 + 6/2);   => 1073741826
```

加、减、乘运算的溢出

●有时计算的最终结果似乎不会溢出，但中间结果可能溢出，这也会导致程序出错

●例：(a+b)/2 未必等于 a/2+b/2

```
printf("%d",(2147483646 + 6)/2);   => -1073741822
printf("%d",2147483646/2 + 6/2);   => 1073741826
```

●解决溢出的办法是尽量使用更高精度的数据类型（两个int进行运算会溢出，用两个 long long 进行运算可能就不会溢出）

除法运算

- 除法的计算结果，类型和操作数中精度高的类型相同

除法运算

- 除法的计算结果，类型和操作数中精度高的类型相同
- 两个整数做除法，结果是商。余数忽略

```
22 / 5  =  4
-22 / 5 = -4
```

除法运算

```
int main()    {
        int a = 10;
        int b = 3;
        double d = a/b; // a/b 的值也是整型，其值是3
        cout << d << endl;  //输出 3
        d = 5/2;            //d的值变为2.0
        cout << d << endl; //输出 2
        d = 5/2.0;
        cout << d << endl; //输出 2.5
        d = (double)a/b;
        cout << d << endl; //输出 3.33333
        return 0;
}
```

模运算

求余数的运算符“%”也称为模运算符。它是双目运算符，两个操作数都是整数类型的。a % b 的值就是a除以b的余数。

22 % 5 = 2
−22 % 5 = −2

模运算

求余数的运算符“%”也称为模运算符。它是双目运算符，两个操作数都是整数类型的。a % b 的值就是a除以b的余数。

22 % 5 = 2
−22 % 5 = −2

除法运算和模运算的除数都不能为0，否则程序会崩溃！！！

自增运算符“++”

- 单目运算符，操作数为整数类型变量或实数型变量

自增运算符“++”

- 单目运算符，操作数为整数类型变量或实数型变量
- 有前置和后置两种用法

自增运算符“++”

- 单目运算符，操作数为整数类型变量或实数型变量
- 有前置和后置两种用法
- 前置用法：

  ++a;　　将a的值加1，表达式返回值为a加1后的值

自增运算符“++”

- 单目运算符，操作数为整数类型变量或实数型变量
- 有前置和后置两种用法
- 前置用法：

  ++a;　　将a的值加1，表达式返回值为a加1后的值

- 后置用法：

  a++;　　将a的值加1，表达式返回值为a加1前的值

自增运算符“++”

```
#include <iostream>
using namespace std;
int main()
{
        int n1 ,  n2 = 5;
        n2 ++;  // n2变成6
        ++ n2;  // n2变成 7
        n1 = n2 ++;  // n2变成8,n1变成7
        cout << n1 << "," << n2 << endl;  //输出 7,8
        n1 = ++ n2;  //n1和n2都变成9
        cout << n1 << "," << n1 << endl;  //输出 9,9
        return 0;
}
```

自减运算符“--”

自减运算符“--”

用于将整型或实数型变量的值减1。它的用法和“++”相同