

第三章 运算符与表达式

主讲：刘晓光

张海威 张莹

殷爱茹 李雨森

宋春瑶 沈玮

卢少平



基本概念 ☐
赋值运算 ☒
算术运算 ☐
关系运算 ☐

☐ 赋值运算符
☐ 赋值表达式求值
☐ 优先级与结合性
☐ 赋值运算中的隐式类型转换



基本概念



赋值运算



算术运算



关系运算



逻辑运算



位运算



条件运算



其它运算



赋值运算符 (assignment operator)

一般赋值运算符: =

复合赋值运算符 (Compound assignment operator)

- 与算术运算符或位运算符进行复合
- +=、-=、*=、/=、%=、>>=、<<=、&=、|=、^=



赋值表达式

<左运算分量> = <右运算分量>

- 左运算分量一般是与表达式值类型相同的变量
- 右运算分量是表达式

<左运算分量> <算术或位运算符> = <右运算分量>

- 等价于 <左运算分量> = <左运算分量> <算术或位运算符> <右运算分量>

运算步骤与表达式求值

运算步骤

- 计算右运算分量的值
 - 表达式的值
- 将该值赋给左运算分量
 - 变量

赋值表达式的值

- 赋值表达式的值是左运算分量的值
 - 赋值运算结束后，右运算分量的值赋给左运算分量，该值同时也是赋值表达式的值



【例】

声明整型变量: int i=1, j=10, k;

- i=j+5;
 - 将表达式j+5的值, 赋值给变量i, 此时变量i的值变为15
- k=i=j+5;
 - 将赋值表达式i=j+5的值, 也即变量i的值15, 进一步赋值给变量k, 使k也等于15
- (j=33)=55;
 - 赋值表达式j=33是左值, 等同于左运算分量j, 可进一步被赋值, 结果将使j的值变为55



声明整型变量: `int i=1, j=10, k;`

- `(j=i=66)++;`

- 结果将使i等于66, 使j的值变为67。注意, “j=i=66”使j等于66, 且该子表达式就等同于变量j, 从而可进一步进行“++”运算

`2=j+5`以及`i+1=55`都是错误的赋值表达式, 因为赋值号的左端非左值



复合赋值运算符

复合赋值运算符举例

- $i+=2$ 等同于 $i=i+2$ ，而 $j\%=i$ 等同于 $j=j\%i$
- 当 $i=10$ 且 $j=3$ 时， $i*=j-2$ 与 $i=i*(j-2)$ 的运算结果都为10

注意，C++认为复合赋值运算符的右运算分量是一个整体，可以理解为编译系统将自动地为右运算分量加上了括号（即是说， $i*=j-2$ 并不等同于 $i=i*j-2$ ，否则的话，将使 i 的值变为28而非10）

X

$$a \% b \\ = a - b * (a / b)$$

优先级与结合性

优先级

- 算术运算符、关系运算符、双目逻辑运算符的优先级均高于赋值运算符

结合性

- 赋值运算符的结合性是自右向左（如， **$k=i=j+5$** 等同于 **$k=(i=j+5)$** ）



赋值过程中的隐式类型转换

能够进行赋值运算的运算分量类型

- 整型=浮点型
- 浮点型=整型
- 单精度浮点型=双精度浮点型
- 双精度浮点型=单精度浮点型
- 整型=字符型
- 字符型=整型
- 无符号=有符号

```
int a;  
a = 3.14;  
  
float a;  
float b;  
double b = 3.14;  
a = b;  
double a;  
float b = 3.14;  
int a;  
char a;  
a = 97;  
cout<<a<<endl;  
unsigned a;  
a = -3;  
cout<<a<<endl;
```

