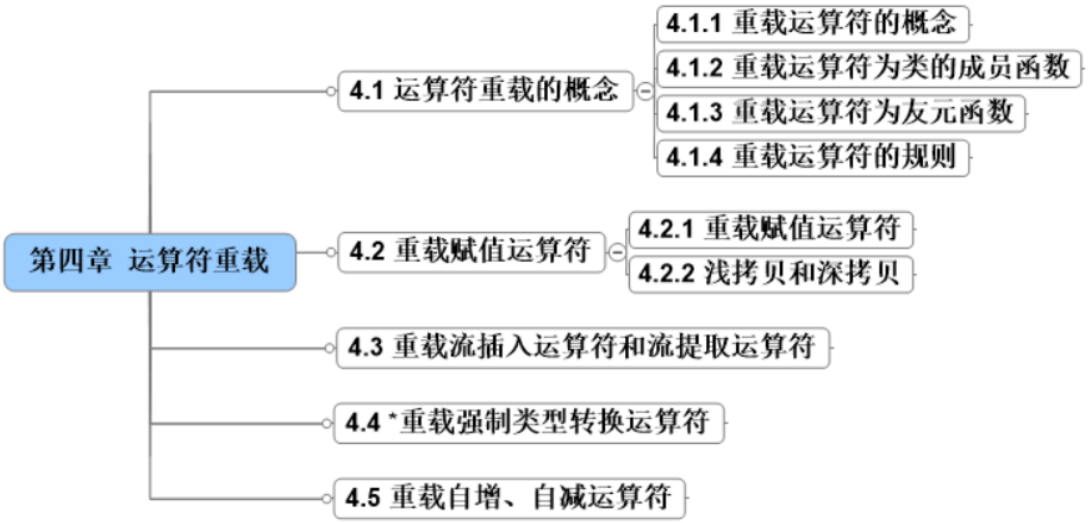
C++程序设计第七节课官方笔记

目录

一、课件下载及重播方法

二、教材节构图



三、本章知识点及考频总结

（一）选择题（共 2 道）

1.运算符重载，就是给已有的运算符赋予多重含义，使同一个运算符作用于不同类型的数据时产生不同的行为。运算符重载的目的是使得 C++中的运算符也能够用来操作对象。C++允许重载大部分的内置运算符。可重载的运算符列在表 4-1 中。

表 4-1 可重载的运算符

双目算术运算符	+(加)，-(减)，*(乘)，/(除)，%(取模)
关系运算符	==(等于)，!=(不等于)，<(小于)，>(大于)，<=(小于等于)，>=(大于等于)
逻辑运算符	||(逻辑或)，&&(逻辑与)，!(逻辑非)
单目运算符	+(正)，-(负)，*(指针)，&(取地址)
自增自减运算符	++(自增)，--(自减)
位运算符	|(按位或)，&(按位与)，~(按位取反)，^(按位异或)，<<(左移)，>>(右移)
赋值运算符	=(赋值)，+=(加法赋值)，-=(减法赋值)，*=(乘法赋值)，/=(除法赋值)，%=(取模赋值)，&=(按位与赋值)，|=(按位或赋值)，^=(按位异或赋值)，<<=(左移赋值)，>>=(右移赋值)
空间申请与释放	new(创建对象)，delete(释放对象)，new[](创建数组)，delete[](释放数组)
其他运算符	()(函数调用)，->(成员访问)，,(逗号)，[](下标)

不可重载的运算符及符号列在表 4-2 中。

成员访问运算符	.
成员指针访问运算符	.*,->*
域运算符	::
长度运算符	sizeof
条件运算符	?:
预处理符号	#

用于类运算的运算符通常都要重载。有两个运算符，系统提供了默认的重载版本。它们是**赋值运算符=和地址运算符&**。对于=，系统默认重载为对象成员变量的复制。对于&，系统默认重载为返回任何类对象的地址。

运算符重载的实质是编写以运算符为名称的函数，使用运算符的表达式就被解释为对重载函数的调用。函数名由**关键字 operator** 和其后要重载的运算符符号构成。与其他函数一样，重载运算符有一个返回类型和一个参数列表。这样的函数称为运算符函数。运算符函数的格式如下：

```
返回值类型 operator 运算符(形参表)
{
    函数体
}
```

运算符可以被重载为全局函数，也可以被重载为类的成员函数。如果定义为全局函数，对于二元运算符，需要为函数传递两个参数，即函数的参数个数就是运算符的操作数个数，运算符的操作数就成为函数的实参。如果定义为类的成员函数，对于二元运算符，则只需要传递一个参数，即函数的参数个数就是运算符的操作数个数减 1，运算符的操作数有一个成为函数作用的对象，其余的成为成员函数的实参。声明为全局函数时，通常应是类的友元。

包含被重载的运算符的表达式会被编译成对运算符函数的调用，运算符的操作数成为函数调用时的实参，运算的结果就是函数的返回值。运算符可以被多次重载。一般来说，倾向于将运算符重载为类的成员函数，这样能够较好地体现运算符和类之间的关系。

2.重载运算符的规则

在 C++中进行运算符重载时，有以下问题需要注意：

1)重载后运算符的含义应该符合原有的用法习惯。
2)运算符重载不能改变运算符原有的语义，包括运算符的优先级和结合性。
3)运算符重载不能改变运算符操作数的个数及语法结构。
4)不能创建新的运算符。
5)重载运算符“()”“[]”“->”或者赋值运算符“=”时，只能将它们重载为成员函数，不能重载为全局函数。
6)运算符重载不能改变该运算符用于基本数据类型对象的含义。

（二）主观题（共 1 道）

为类 myComplex 重载运算符“+”和“-”

```
#include<iostream>
using namespace std;
class myComplex  //复数类
{
private:
    double real,imag;
public:
    myComplex();                                //构造函数
    myComplex(double r,double i);               //构造函数
    void outCom();                               //成员函数
    myComplex operator-(const myComplex &c);  //成员函数
    friend myComplex operator+(const myComplex &c1,const myComplex &c2);
};
myComplex::myComplex()
{
    real=0;
    imag=0;
}
myComplex::myComplex(double r,double i)
{
    real=r;
    imag=i;
}
void myComplex::outCom()
{
    cout<<"("<<real<<","<<imag<<")";
}
myComplex myComplex::operator - (const myComplex &c)
{
  return myComplex(this->real - c.real,this->imag - c.imag); //返回一个临时对象
}
myComplex operator+(const myComplex &c1,const myComplex &c2)
{
    return myComplex(c1.real+c2.real,c1.imag+c2.imag);       //返回一个临时对象
```

```
}
int main()
{
    myComplex c1(1,2),c2(3,4),res;
    c1.outCom();
    cout<<"operator+";
    c2.outCom();
    cout<<"=";
    res=c1+c2;
    res.outCom();
    cout<<endl;
    c1.outCom();
    cout<<"operator-";
    c2.outCom();
    cout<<"=";
    res=c1-c2;
    res.outCom();
    cout<<endl;
    return 0;
}
```

程序 4-1 的执行结果如下:

```
(1,2)operator+(3,4)=(4,6)
(1,2)operator-(3,4)=(-2,-2)
```

【程序说明】类 myComplex 中重载了两个运算符，为了对比，operator-重载为成员函数，而 operator+重载为友元函数。重载为成员函数时最方便，而且传递的参数个数比运算符需要的操作数个数少一个，因为调用对象本身也参与运算，例如，operator-的函数体中使用 this 表示对象本身，当然在这里 this 完全可以省略。

四、配套练习题

1、下列关于运算符重载的叙述，正确的是（　　）

A:通过运算符重载，可以定义新的运算符

B:有的运算符只能作为成员函数重载

C:若重载运算符+，则相应的运算符函数名是+

D:重载一个二元运算符时，必须声明两个形参

2、下列运算符中，在 C++中不能重载的是（　　）

A:+

B:>=

C:::

D:/

3、以下关于运算符重载的描述中，错误的是（　）

A:运算符重载其实就是函数重载

B:成员运算符比友元运算符少一个参数

C:需要使用关键字 operator

D:成员运算符比友元运算符多一个参数

[参考答案] BCD

五、其余课程安排