Chap03 重载和类型转换

程勇

北京化工大学

信息科学与技术学院计算机系

Sept. 2020

课程提纲

- Chap01 绪论
- Chap02 面向对象程序设计方法
- Chap03 重载与类型转换
- Chap04 继承与派生
- Chap05 多态性
- Chap06 输入输出流
- Chap07 容错与异常处理
- Chap08 模版

本章提纲

- 函数重载
  - 构造函数重载
  - 函数重载
- 运算符重载
  - 可重载的运算符
  - 运算符重载规则
- 运算符重载实现
  - 重载赋值运算符
  - 重载算术运算符
- 数据类型转换

基本概念

- 比较两个盒子的体积
  - if(box1.compareVolume(box2))
  - if(box1 < box2)　// 运算符重载
- 重载运算符
  - 基本格式

  返回值类型 类名::operator重载的运算符(参数列表){

  }
  - operator是关键字，它与重载的运算符一起构成函数名，C++编译器可以将这类函数识别出来；
  - 为指定类重载运算符的函数不一定是该类的成员，可以是一般函数，如全局函数等；
- 除下列运算符之外，其他运算符都是可以被重载的
  - 作用域运算符(::)、成员与成员指针操作符(.和.*)、sizeof运算符(sizeof)、条件运算符(? :)

关系运算符重载(成员函数)

- 重载<运算符

  if(box1<box2)

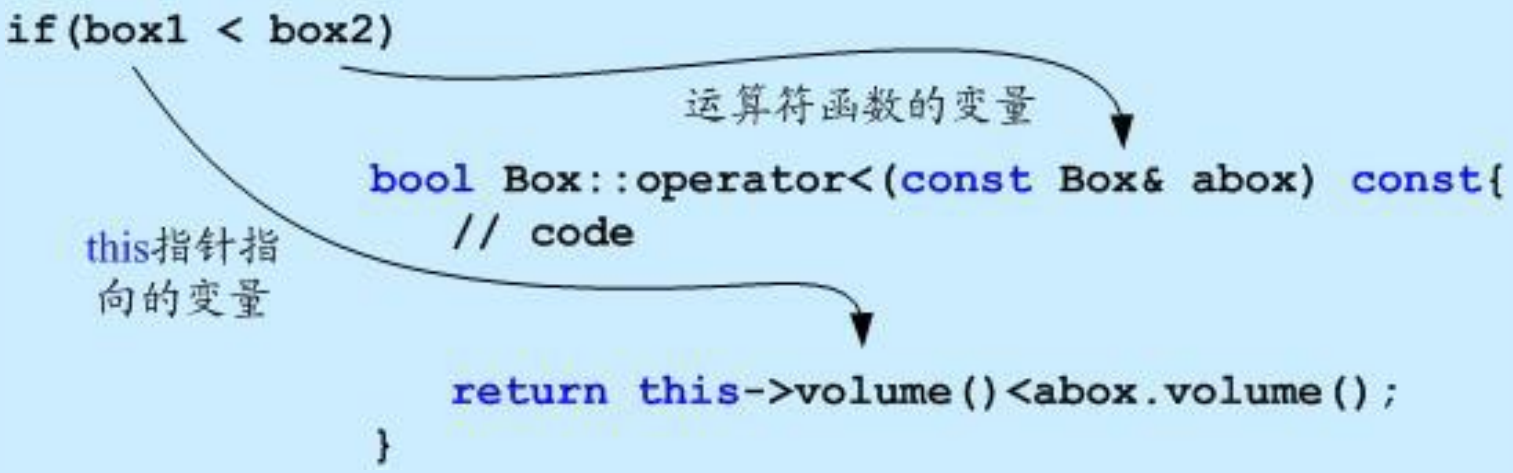
  cout<< “box1 is less than box2” << endl;

  - 该表达式等价于函数调用

  if(box1.operator<(box2))

  cout<< “box1 is less than box2” << endl;

- 映射关系



关系运算符重载(全局函数)

- 重载<运算符

  if(box1<box2)

  cout<< “box1 is less than box2” << endl;

  - 该表达式等价于函数调用

  if(operator<(box1, box2))

  cout<< “box1 is less than box2” << endl;

- 映射关系

```
if(box1 < box2)
```

运算符函数的变量

```
bool operator<(const Box& box1, const Box& box2){
    // code

    return box1.volume()<box2.volume();
}
```

重载赋值运算符

- 赋值运算符将给定类型的对象复制到同一类型的另一个对象中
  - 返回类型必须是该对象的引用；
  - 比较两个操作数是否相等；
  - 删除左操作数拥有的所有对象；
  - 进行复制操作(动态分配内存)；
- 实现赋值运算符
  - load1=load2=load3；等价于

load1.operator=(load2.operator=(load3));

- 注意事项
  - 副本构造函数和赋值运算符是完全不同的，不要将两者混淆：
    - 在类对象用该类的另一个已有对象进行创建和初始化时，或者对象按值传送给函数时，调用副本构造函数；
    - 当赋值语句左边和右边是同一对象时，才调用赋值运算符函数；
  - 如果类的函数在自由存储区中动态分配内存，就总是应该实现副本构造函数，赋值运算符和析构函数；

重载算术运算符

- 算术运算符计算的结果往往得到一个新的对象
  - 确定算术运算符号的含义；
  - 创建本地对象，并返回副本；
- 实现算术运算符
  - Box box4 = box1 + box2 + box3;

  成员函数重载

  Box operator+(const Box& abox) const;

  一般函数重载

  Box operator+(const Box& abox, const Box& bbox);
- 注意事项
  - 实现符合算术运算符 (+=、-=、*=、/=)时，需要返回一个引用；

  Box& operator+=(const Box& right);
  - 可以根据一个运算符定义另一个运算符，如由+=来实现+等等；

重载下标运算符

- 下标运算符含义
  - 从许多可解释为数组(如稀疏矩阵、链表、文件等)的对象中进行选择；
  - 可以改善标准下标运算符的工作方式，如检查索引值的合法性；
- 实现下标运算符
  - load[3]；等价于

```
Box operator[](int index) const;
```

- 注意事项
  - 下标运算符为赋值语句左边的值，即load[0] = load[1]，此时重载函数应该返回一个引用：

```
Box& operator[](int index) const;
```

  - const上的重载，即在某些情况下不允许对返回的对象进行修改，这时可定义重载函数为：

```
const Box& operator[](int index) const;
```

重载递增和递减运算符

- 递增和递减运算符
  - 每个运算符需要两个函数以区分前缀和后缀表达式；
- 实现递增和递减运算符
  - object++; 等价于

  const Object operator++(int);
  - ++object; 等价于

  Object& operator++();
  - object--; 等价于

  const Object operator--(int);
  - --object; 等价于

  Object& operator--();
- 注意事项
  - 重载运算符的任何实现，前缀形式的返回类型是当前对象的引用，后缀形式的返回对象是同一类型的新对象，该对象是递增操作执行之前的原对象的副本；

本章小结

- 函数重载
- 运算符重载
- 运算符重载实现
- 数据类型转换

- 课程网站
  - http://www.jiaowu.buct.edu.cn