1. 程序代码

程序一

#include<iostream>

using namespace std;

class Complex {

public:

double real;

double imag;

Complex(double r = 0, double i = 0)

{

real = r; imag = i;

}

};

Complex operator+(Complex co1, Complex co2)

{

Complex temp;

temp.real = co1.real + co2.real;

temp.imag = co1.imag + co2.imag;

return temp;

}

int main()

{

Complex com1(1.1, 2.2), com2(3.3, 4.4), total1, total2;

total1 = operator+(com1, com2);

cout << "real1=" << total1.real << " " << "imag1=" << total1.imag << endl;

total2 = com1 + com2;

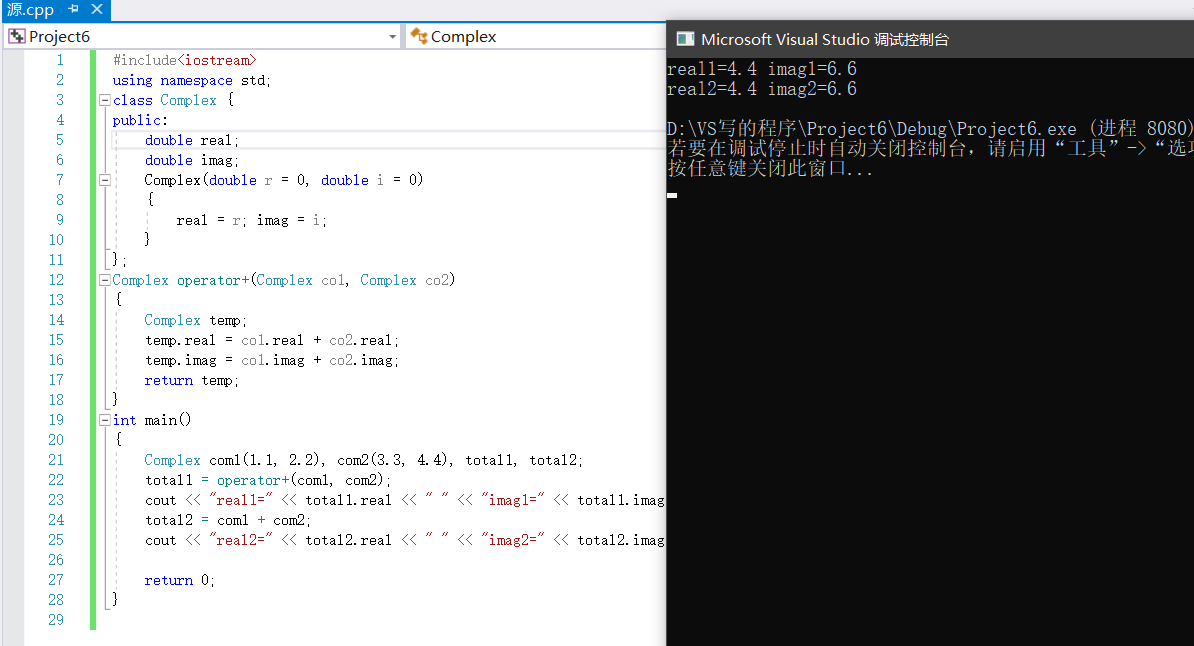
cout << "real2=" << total2.real << " " << "imag2=" << total2.imag << endl;

return 0;

}

1. 运行结果

程序一



1. 心得和感想

程序一  
1.运算符重载时要遵循以下规则：  
(1) 除了类属关系运算符"."、成员指针运算符".\*"、作用域运算符"::"、sizeof运算符和三目运算符"?:"以外，C++中的所有运算符都可以重载。  
(2) 重载运算符限制在C++语言中已有的运算符范围内的允许重载的运算符之中，不能创建新的运算符。  
(3) 运算符重载实质上是函数重载，因此编译程序对运算符重载的选择，遵循函数重载的选择原则。  
(4) 重载之后的运算符不能改变运算符的优先级和结合性，也不能改变运算符操作数的个数及语法结构。  
(5) 运算符重载不能改变该运算符用于内部类型对象的含义。它只能和用户自定义类型的对象一起使用，或者用于用户自定义类型的对象和内部类型的对象混合使用时。