## 实验报告七

Copyright ©2021-2099 YalanLi. All rights reserved

## 程序代码：

#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 - co1.imag \* co2.imag;

temp.imag = co1.imag \* co2.real + co2.imag \* co1.real;

return temp;

}

int main()

{

cout << "请分别输入两个复数的实部和虚部。" << endl;

double x[2], y[2];

cin >> x[0] >> y[0] >> x[1] >> y[1];

Complex com1(x[0], y[0]), com2(x[1], y[1]), total;

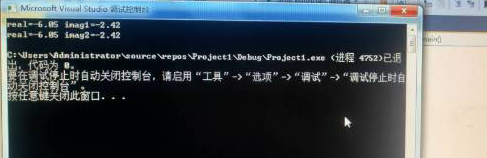
total = com1 \* com2;

cout << "com1(1.1, 2.2) \* com2(3.3, 4.4) = real（" << total.real << " " << "）+ imag（" << total.imag << "）" << endl;

return 0;

}

## 运行结果：



## 感想心得：

我深刻认识到了运算符重载的好处：从开发的角度而言，运算符重载的存在是为了提高开发效率，增加代码的可复用性，很多时候反而是简化了问题。自定义的数据类型通过操作符重载具有了和内建类型同样的使用接口，然后C++的模板加持下，你的算法可以利用这种接口一致性，实现泛化。