#include<iostream>

using namespace std;

class Complex {

double real;

double imag;

public:

Complex(double r = 0.0, double i = 0.0)

{

real = r;

imag = i;

}

void print();

friend Complex operator\*(Complex &a, Complex &b);

};

Complex operator\*(Complex &a, Complex &b)

{

Complex temp;

temp.real = a.real \* b.real-a.imag\*b.imag;

temp.imag = a.real \* b.imag+a.imag\*b.real;

return temp;

}

void Complex::print()

{

cout << real;

if (imag > 0)

{

cout << "+";

}

if (imag != 0)

{

cout << imag << 'i' << endl;

}

//cout << "total real=" << real << " " << "total imag=" << imag << endl;

}

int main()

{

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

total1 = com1 \* com2;

total1.print();

return 0;

}

实验结果：



**实验总结**

运算符重载：

1.运算符重载是通过创建运算符重载函数来实现的，运算符重载函数可以是在类外定义的普通函数，也可以是类的成员函数或友元函数。

C++为运算符重载提供了一种方法，即在进行运算符重载时，必须定义一个运算符重载函数，其名字为operator，后随一个要重载的运算符。

2.C++中绝大多数的运算符允许重载，不能重载的运算符有以下几个：

. 成员访问运算符；

.\* 成员指针访问运算符；

:: 作用域运算符；

Sizeof 长度运算符；

?: 条件运算符；

3，C++语言只能已有的运算符进行重载，不允许用户自己定义运算符；