程序代码

#include<iostream>

using namespace std;

class A

{

public:

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

{

real = r;

imag = i;

}

double real, imag;

};

A operator\*(A co1, A co2)

{

A t;

t.real = co1.real\*co2.real - co1.imag\*co2.imag;

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

return t;

}

int main()

{

A com1(2.3, 4.6), com2(3.6, 2.8),t1;

t1 = com1 \* com2;

cout << "t1 real=" << t1.real << " " << "t1 imag=" << t1.imag << endl;

t1 = operator\* (com1,com2);

cout << "t1 real=" << t1.real << " " << "t1 imag=" << t1.imag << endl;

}

运行结果



感想心得

运算符重载的概念，就是把运算符当作多态函数，对它们进行重新定义，并赋予新的功能。就是说运算符不仅是一个运算符了，被赋予了新的意义，图中代码便是将\*运算符定义成了复数的乘法。其中代码的 :: 为作用域运算符，要注意在进行友元函数定义时，在类外定义友元函数，不需要“类名::”。另外对双目运算符而言，成员运算符重载函数的形参表中仅有一个参数，是运算符的右操作数。而其中对于单目运算符，建议选择成员函数；对于运算符“=、（）、[]、->”只能作为成员函数；对于运算符“+=、-=、/=、\*=、&=、！=、~=、%=、<<= 、>>=”，建议重载为成员函数；对于其他运算符，建议重载为友元函数。