编写一个程序，实现两个复数的相加

#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;

}

作业：尝试编写一个程序，实现两个复数的乘法

#include<iostream>

using namespace std;

class Complex {

public:

double real;

double image;

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

real = r;

image = i;

}

};

Complex operator\*(Complex co1, Complex co2) {

Complex temp;

temp.real = co1.real \* co2.real - co1.image \* co2.image;

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

return temp;

}

int main() {

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

total1 = operator\*(com1, com2)；

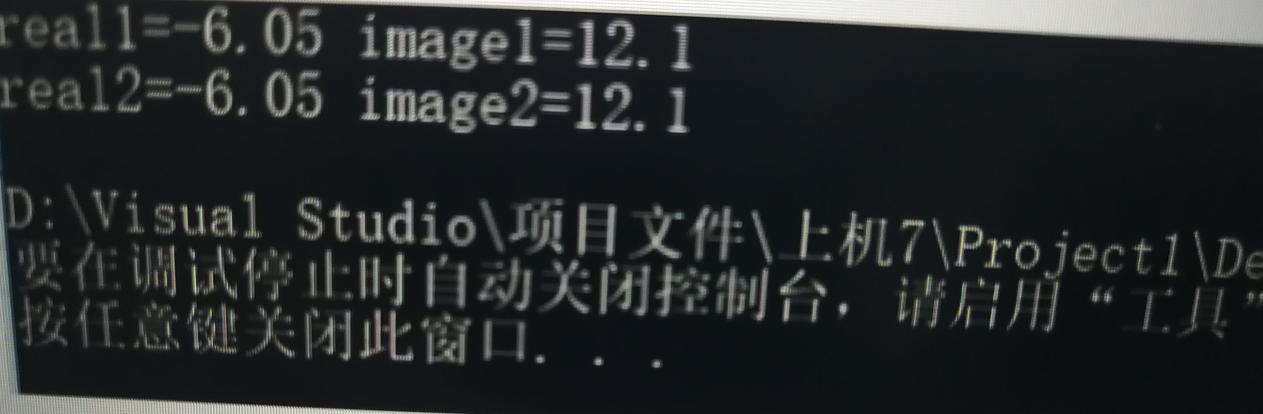
cout << "real1=" << total1.real <<" " << "image1=" << total1.image << endl;

total2 = com1 \* com2;

cout << "real2=" << total2.real << " " << "image2=" << total2.image << endl;

return 0;

}



上机心得：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，它作为运算符的右操作数。

另一个操作数（左操作数）是隐含的，是该类的当前对象，他是通过this指针隐含传递给函数。

乘法比加法运算方式略有不同，实质变化不大。

Copyright IMG_2562021-2099 wangzezheng. All rights reserved