Main.cpp

// Preston Knibbe

#include <iostream>

#include "ComplexNumber.h"

using namespace std;

int main()

{

ComplexNumber Number1(10, 10);

ComplexNumber Number2(5, 9);

ComplexNumber Result;

cout << "Number 1: " << endl;

Number1.Print();

cout << endl;

cout << "Number 2: " << endl;

Number2.Print();

cout << endl;

Result.Addition(Number1, Number2);

cout << "Addition total: " << endl;

Result.Print();

cout << endl;

Result.Subtraction(Number1, Number2);

cout << "Subtraction result: " << endl;

Result.Print();

cout << endl;

Result.Multiplication(Number1, Number2);

cout << "Multiplication total: " << endl;

Result.Print();

cout << endl;

Result.Division(Number1, Number2);

cout << "Division result: " << endl;

Result.Print();

return 0;

}

ComplexNumber.cpp

// Preston Knibbe

#include <iostream>

#include "ComplexNumber.h"

#include "math.h"

using namespace std;

ComplexNumber::ComplexNumber() {

real\_part = 0;

imaginary\_part = 0;

}

ComplexNumber::ComplexNumber(double real, double imaginary)

{

setCompNum(real, imaginary);

}

void ComplexNumber::Print() {

cout << "Real part: " << real\_part << endl;

cout << "Imaginary part: " << imaginary\_part << endl;

}

void ComplexNumber::Addition(ComplexNumber add1, ComplexNumber add2) {

real\_part = add1.getRealPart() + add2.getRealPart();

imaginary\_part = add1.getImaginaryPart() + add2.getImaginaryPart();

}

void ComplexNumber::Subtraction(ComplexNumber subtr1, ComplexNumber subtr2) {

real\_part = subtr1.getRealPart() - subtr2.getRealPart();

imaginary\_part = subtr1.getImaginaryPart() - subtr2.getImaginaryPart();

}

void ComplexNumber::Multiplication(ComplexNumber mult1, ComplexNumber mult2) {

real\_part = (mult1.getRealPart() \* mult2.getRealPart()) - (mult1.getImaginaryPart() \* mult2.getImaginaryPart());

imaginary\_part = (mult1.getRealPart() \* mult2.getImaginaryPart()) + (mult1.getImaginaryPart() \* mult2.getRealPart());

}

void ComplexNumber::Division(ComplexNumber div1, ComplexNumber div2) {

real\_part = ((div1.getRealPart()\*div2.getRealPart()) + (div1.getImaginaryPart()\*div2.getImaginaryPart())) / ((div2.getImaginaryPart() \* div2.getImaginaryPart()) + (div2.getRealPart()\*div2.getRealPart()));

imaginary\_part = ((div2.getRealPart()\*div1.getImaginaryPart())-(div1.getRealPart()\*div2.getImaginaryPart()))/((div2.getRealPart()\*div2.getRealPart())+(div2.getImaginaryPart()\*div2.getImaginaryPart()));

}

void ComplexNumber::Polar(ComplexNumber pol) {

cout << "Magnitude: " << sqrt((pol.getRealPart()\*pol.getRealPart())+(pol.getImaginaryPart()\*pol.getImaginaryPart())) << endl;

cout << "Angle: " << atan(pol.getImaginaryPart()/pol.getRealPart()) << endl;

}

double ComplexNumber::getRealPart() {

return real\_part;

}

double ComplexNumber::getImaginaryPart() {

return imaginary\_part;

}

void ComplexNumber::setCompNum(double real, double imaginary) {

real\_part = real;

imaginary\_part = imaginary;

}

ComplexNumber.h

// Preston Knibbe

#ifndef COMPLEXNUMBER\_H

#define COMPLEXNUMBER\_H

#include <iostream>

class ComplexNumber

{

public:

ComplexNumber();

ComplexNumber(double real, double imaginary);

void Print();

void Addition(ComplexNumber add1, ComplexNumber add2);

void Subtraction(ComplexNumber subtr1, ComplexNumber subtr2);

void Multiplication(ComplexNumber mult1, ComplexNumber mult2);

void Division(ComplexNumber div1, ComplexNumber div2);

void Polar(ComplexNumber pol);

double getRealPart();

double getImaginaryPart();

void setCompNum(double real, double imaginary);

private:

double real\_part;

double imaginary\_part;

};

#endif // COMPLEXNUMBER\_H

Output

Number 1:

Real part: 10

Imaginary part: 10

Number 2:

Real part: 5

Imaginary part: 9

Addition total:

Real part: 15

Imaginary part: 19

Subtraction result:

Real part: 5

Imaginary part: 1

Multiplication total:

Real part: -40

Imaginary part: 140

Division result:

Real part: 1.32075

Imaginary part: -0.377358

Process returned 0 (0x0) execution time : 0.016 s

Press any key to continue.

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Number 1:

Real part: 6

Imaginary part: 8

Number 2:

Real part: 4

Imaginary part: 3

Addition total:

Real part: 10

Imaginary part: 11

Subtraction result:

Real part: 2

Imaginary part: 5

Multiplication total:

Real part: 0

Imaginary part: 50

Division result:

Real part: 1.92

Imaginary part: 0.56

Process returned 0 (0x0) execution time : 0.073 s

Press any key to continue.

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Number 1:

Real part: 3

Imaginary part: 7

Number 2:

Real part: 50

Imaginary part: 2

Addition total:

Real part: 53

Imaginary part: 9

Subtraction result:

Real part: -47

Imaginary part: 5

Multiplication total:

Real part: 136

Imaginary part: 356

Division result:

Real part: 0.0654952

Imaginary part: 0.13738

Process returned 0 (0x0) execution time : 0.014 s

Press any key to continue.