**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Объектно-ориентированное программирование»

**Лабораторная работа № 1**

Тема: Простые классы на языке С++

Студент: Мариничев Иван Александрович

Группа: 80-208

Преподаватель: Чернышов Л.Н.

Дата: 28.09.20

Оценка:

Москва, 2020

1. Постановка задачи

**Комплексное число в алгебраической форме** представляются парой действительных чисел (a, b), где a – действительная часть, b – мнимая часть. Реализовать класс Complex для работы с комплексными числами. Обязательно должны быть присутствовать операции

- сложения add, (a, b) + (c, d) = (a + c, b + d);

- вычитания sub, (a, b) – (c, d) = (a – c, b – d);

- умножения mul, (a, b) ´ (c, d) = (ac – bd, ad + bc);

- деления div, (a, b) / (c, d) = (ac + bd, bc – ad) / (c2 + d2);

- сравнение equ, (a, b) = (c, d), если (a = c) и (b = d);

- сопряженное число conj, conj(a, b) = (a, –b).

Реализовать операции сравнения модулей.

1. Описание программы

Программа предназначена для работы с комплексными числами. В данной программе реализованы операции сложения, вычитания, умножения и деления комплексных чисел. Также данная программа может проверять равенство комплексных чисел, а также сравнивать модули комплексных чисел.

1. Набор тестов

**test\_00:**

1 3 0 0

**test\_01:**

0 0 0 0

**test\_02:**

3 4 5 6

**test\_03:**

-2 -3 -2 -3

**test\_04:**

15 65 -30 - 70

**test\_09:**

Несуществующий файл

1. Результаты выполнения тестов

**test\_00:**

Complex 1: (1, 3)

Complex 2: (0, 0)

Sum of complex numbers: (1, 3)

Difference of complex numbers: (1, 3)

Product of complex numbers: (0, 0)

Can't divide those complex numbers

Complex 1 conjugate: (1, -3)

Complex 2 conjugate: (0, -0)

Comparing absolute values of 2 complex numbers...

Absolute value of Complex 1 is not equal to Absolute value of Complex 2

Absolute value of Complex 1 is greater than Absolute value of Complex 2

As an addition we can add and subtract real numbers

Let's add 8 to Complex 1:

(9, 3)

Let's subtract 15 from Complex 2:

(-15, 0)

**test\_01:**

Complex 1: (0, 0)

Complex 2: (0, 0)

Sum of complex numbers: (0, 0)

Difference of complex numbers: (0, 0)

Product of complex numbers: (0, 0)

Can't divide those complex numbers

Complex 1 conjugate: (0, -0)

Complex 2 conjugate: (0, -0)

Comparing absolute values of 2 complex numbers...

Absolute value of Complex 1 is equal to Absolute value of Complex 2

As an addition we can add and subtract real numbers

Let's add 8 to Complex 1:

(8, 0)

Let's subtract 15 from Complex 2:

(-15, 0)

**test\_02:**

Complex 1: (3, 4)

Complex 2: (5, 6)

Sum of complex numbers: (8, 10)

Difference of complex numbers: (-2, -2)

Product of complex numbers: (-9, 38)

Quotient of complex numbers: (0.639344, 0.0327869)

Complex 1 conjugate: (3, -4)

Complex 2 conjugate: (5, -6)

Comparing absolute values of 2 complex numbers...

Absolute value of Complex 1 is not equal to Absolute value of Complex 2

Absolute value of Complex 1 is lower than Absolute value of Complex 2

As an addition we can add and subtract real numbers

Let's add 8 to Complex 1:

(11, 4)

Let's subtract 15 from Complex 2:

(-10, 6)

**test\_03:**

Complex 1: (-2, -3)

Complex 2: (-2, -3)

Sum of complex numbers: (-4, -6)

Difference of complex numbers: (0, 0)

Product of complex numbers: (-5, 12)

Quotient of complex numbers: (1, 0)

Complex 1 conjugate: (-2, 3)

Complex 2 conjugate: (-2, 3)

Comparing absolute values of 2 complex numbers...

Absolute value of Complex 1 is equal to Absolute value of Complex 2

As an addition we can add and subtract real numbers

Let's add 8 to Complex 1:

(6, -3)

Let's subtract 15 from Complex 2:

(-17, -3)

**test\_04:**

Complex 1: (15, 65)

Complex 2: (-30, 0)

Sum of complex numbers: (-15, 65)

Difference of complex numbers: (45, 65)

Product of complex numbers: (-450, -1950)

Quotient of complex numbers: (-0.5, -2.16667)

Complex 1 conjugate: (15, -65)

Complex 2 conjugate: (-30, -0)

Comparing absolute values of 2 complex numbers...

Absolute value of Complex 1 is not equal to Absolute value of Complex 2

Absolute value of Complex 1 is greater than Absolute value of Complex 2

As an addition we can add and subtract real numbers

Let's add 8 to Complex 1:

(23, 65)

Let's subtract 15 from Complex 2:

(-45, 0)

**test\_09:**

File is not exists

1. Листинг программы

**CmakeLists.txt**

cmake\_minimum\_required(VERSION 3.17)

project(oop\_lab1)

set(CMAKE\_CXX\_STANDARD 17)

file(COPY ${CMAKE\_CURRENT\_SOURCE\_DIR}/test\_00.txt

DESTINATION ${CMAKE\_CURRENT\_BINARY\_DIR})

file(COPY ${CMAKE\_CURRENT\_SOURCE\_DIR}/test\_01.txt

DESTINATION ${CMAKE\_CURRENT\_BINARY\_DIR})

file(COPY ${CMAKE\_CURRENT\_SOURCE\_DIR}/test\_02.txt

DESTINATION ${CMAKE\_CURRENT\_BINARY\_DIR})

file(COPY ${CMAKE\_CURRENT\_SOURCE\_DIR}/test\_03.txt

DESTINATION ${CMAKE\_CURRENT\_BINARY\_DIR})

file(COPY ${CMAKE\_CURRENT\_SOURCE\_DIR}/test\_04.txt

DESTINATION ${CMAKE\_CURRENT\_BINARY\_DIR})

add\_executable(oop\_exercise\_01 main.cpp complex.cpp complex.h)

**main.cpp**

/\* Мариничев Иван М8О-208Б-19

\*

\* github: IvaMarin

\*

\*Реализовать класс Complex для работы с комплексными числами. Обязательно должны быть присутствовать операции

\* -сложения add, (a, b) + (c, d) = (a + c, b + d);

\* -вычитания sub, (a, b) – (c, d) = (a – c, b – d);

\* -умножения mul, (a, b) ´ (c, d) = (ac – bd, ad + bc);

\* -деления div, (a, b) / (c, d) = (ac + bd, bc – ad) / (c2 + d2);

\* -сравнение equ, (a, b) = (c, d), если (a = c) и (b = d);

\* -сопряженное число conj, conj(a, b) = (a, –b).

\*Реализовать операции сравнения модулей.

\*/

#include <iostream>

#include <fstream>

#include "complex.h"

using namespace std;

int main () {

cout << "Creating 2 complex class objects..." << endl;

string test\_name;

cout << "Type test file name (e.g. test\_01.txt) ";

cout << "or type 'console' to enter them by yourself: ";

cin >> test\_name;

double real\_a, imaginary\_b, real\_c, imaginary\_d;

if (test\_name != "console") {

ifstream input;

input.open(test\_name);

if (!input.is\_open()) {

cout << "File is not exists\n"; // если не открылся

return -1;

}

input >> real\_a >> imaginary\_b >> real\_c >> imaginary\_d;

}else {

cout << "Enter real and imaginary parts of Complex 1: ";

cin >> real\_a >> imaginary\_b;

cout << "Enter real and imaginary parts of Complex 2: ";

cin >> real\_c >> imaginary\_d;

}

cout << endl;

Complex c1(real\_a, imaginary\_b);

Complex c2(real\_c, imaginary\_d);

cout << "Complex 1: ";

c1.Display();

cout << "Complex 2: ";

c2.Display();

cout << endl;

Complex sum = c1 + c2;

Complex dif = c1 - c2;

Complex mul = c1 \* c2;

Complex div = c1 / c2 ;

double r = c2.getReal() \* c2.getReal() + c2.getImaginary() \* c2.getImaginary();

cout << "Sum of complex numbers: ";

sum.Display();

cout << "Difference of complex numbers: ";

dif.Display();

cout << "Product of complex numbers: ";

mul.Display();

if (r != 0) {

cout << "Quotient of complex numbers: ";

div.Display();

} else {

cout << "Can't divide those complex numbers";

}

cout << endl;

Complex c1conj = c1.conj();

Complex c2conj = c2.conj();

cout << "Complex 1 conjugate: ";

c1conj.Display();

cout << "Complex 2 conjugate: ";

c2conj.Display();

cout << endl;

cout << "Comparing absolute values of 2 complex numbers..." << endl;

if (c1 != c2) {

cout << "Absolute value of Complex 1 is not equal to Absolute value of Complex 2" << endl;

if (c1 > c2) {

cout << "Absolute value of Complex 1 is greater than Absolute value of Complex 2" << endl;

}

else {

cout << "Absolute value of Complex 1 is lower than Absolute value of Complex 2" << endl;

}

}

else {

cout << "Absolute value of Complex 1 is equal to Absolute value of Complex 2" << endl;

}

cout << endl;

cout << "As an addition we can add and subtract real numbers" << endl;

Complex realadd = c1 + 8;

Complex realsub = c2 - 15;

cout << "Let's add 8 to Complex 1: " << endl;

realadd.Display();

cout << "Let's subtract 15 from Complex 2: " << endl;

realsub.Display();

}

**complex.cpp**

#include "complex.h"

#include <cmath>

/\*

ostream &operator<<(ostream &out, const Complex &c) {

out << "(" << c.getReal() << ", " << c.getImaginary() << ")" << endl;

return out;

}

\*/

Complex operator+(const Complex &c1, const Complex &c2) { //оператор сложения комплексных чисел

return Complex(c1.getReal()+c2.getReal(), c1.getImaginary()+c2.getImaginary());

}

Complex operator+(const Complex &c1, double d) { //оператор сложения комплексного числа и действительного

return Complex(c1.getReal()+d, c1.getImaginary());

}

Complex operator+( double d, const Complex &c1) { //оператор сложения действительного числа и комплексного

return Complex(c1.getReal()+d, c1.getImaginary());

}

Complex operator-(const Complex &c1, const Complex &c2) { //оператор вычитания комплексных чисел

return Complex(c1.getReal()-c2.getReal(), c1.getImaginary()-c2.getImaginary());

}

Complex operator-(const Complex &c1, double d) { //оператор вычитания комплексного числа и действительного

return Complex(c1.getReal()-d, c1.getImaginary());

}

Complex operator-( double d, const Complex &c1) { //оператор вычитания действительного числа и комплексного

return Complex(d-c1.getReal(), c1.getImaginary());

}

Complex operator\*(const Complex &c1, const Complex &c2) {

return Complex(c1.getReal()\*c2.getReal()-c1.getImaginary()\*c2.getImaginary(), c1.getReal()\*c2.getImaginary()+c1.getImaginary()\*c2.getReal());

}

Complex operator/(const Complex &c1, const Complex &c2) {

double r = c2.getReal() \* c2.getReal() + c2.getImaginary() \* c2.getImaginary();

return Complex((c1.getReal()\*c2.getReal()+c1.getImaginary()\*c2.getImaginary()) / r, (c1.getImaginary()\*c2.getReal()-c1.getReal()\*c2.getImaginary()) / r);

}

bool operator==(const Complex &c1, const Complex &c2) {

return (c1.getReal() == c2.getReal() && c1.getImaginary() == c2.getImaginary());

}

bool operator!=(const Complex &c1, const Complex &c2) {

return !(c1 == c2);

}

bool operator<(const Complex &c1, const Complex &c2) {

return (sqrt(c1.getReal()\*c1.getReal() + c1.getImaginary()\*c1.getImaginary()) < sqrt(c2.getReal()\*c2.getReal() + c2.getImaginary()\*c2.getImaginary()));

}

bool operator<=(const Complex &c1, const Complex &c2) {

return (c1 < c2) || (c1 == c2);

}

bool operator>(const Complex &c1, const Complex &c2) {

return c2 < c1;

}

bool operator>=(const Complex &c1, const Complex &c2) {

return c2 <= c1;

}

Complex Complex::conj() {

return Complex(real, -(imaginary));

}

void Complex::Display() {

cout << "(" << real << ", " << imaginary << ")" << endl;

}

Complex::Complex(): real(0), imaginary(0) {} //Конструктор по умолчанию

Complex::Complex(double real, double imaginary): real(real), imaginary(imaginary) {}

Complex::Complex(const Complex &other) {

real = other.real;

imaginary = other.imaginary;

} //конструктор копирования

const Complex &Complex::operator=(const Complex &other) {

real = other.real;

imaginary = other.imaginary;

return \*this;

}

**complex.h**

#ifndef COMPLEX\_H\_

#define COMPLEX\_H\_

#include <iostream>

using namespace std;

class Complex

{

private:

double real; //действительная часть комплексного числа

double imaginary; //мнимая часть комплексного числа

public:

Complex();

Complex(double real, double imaginary);

Complex(const Complex &other); //конструктор копирования

const Complex &operator=(const Complex &other); //оператор присваивания

double getReal() const {

return real;

}

double getImaginary() const {

return imaginary;

}

Complex conj();

void Display();

};

//ostream &operator<<(ostream &out, const Complex &c);

Complex operator+(const Complex &c1, const Complex &c2);

Complex operator+(const Complex &c1, double d);

Complex operator+(double d, const Complex &c1);

Complex operator-(const Complex &c1, const Complex &c2);

Complex operator-(const Complex &c1, double d);

Complex operator-(double d, const Complex &c1);

Complex operator\*(const Complex &c1, const Complex &c2);

Complex operator/(const Complex &c1, const Complex &c2);

bool operator==(const Complex &c1, const Complex &c2);

bool operator!=(const Complex &c1, const Complex &c2);

bool operator<(const Complex &c1, const Complex &c2);

bool operator<=(const Complex &c1, const Complex &c2);

bool operator>(const Complex &c1, const Complex &c2);

bool operator>=(const Complex &c1, const Complex &c2);

#endif /\* COMPLEX\_H\_ \*/