МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСТИТЕТ)

ЛАБОРАТОРНАЯ РАБОТА №1

по курсу объектно-ориентированное программирование I семестр, 2021/22 уч. год

Студент *Абаровский Олег Александрович, группа М8О-207Б-20*

Преподаватель *Дорохов Евгений Павлович*

Условие

Разработать программу на языке C++ согласно варианту задания. Программа на C++ должна собираться с помощью системы сборки CMake. Программа должна получать данные из стандартного ввода и выводить данные в стандартный вывод.
Необходимо настроить сборку лабораторной работы с помощью CMake. Собранная программа должна называться oop_exercise_01
Задание:
Вариант 1: 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) / (c 2 + d 2);
- сравнение equ, (a, b) = (c, d), если (a = c) и (b = d);
- сопряженное число conj, conj(a, b) = (a, –b).

Описание программы

Исходный код лежит в файле main.cpp, в котором реализован класс комплексного числа и описаны методы этого класса, среди которых сложение, вычитание, умножение, деление, сравнение, нахождение сопряженного числа.

Дневник отладки

Исправлений не потребовалось.

Недочёты

Выводы

В ходе выполнения лабораторной работы я изучил реализацию классов в c++ и создал класс с набором методов согласно варианту задания. Работа была полезной, так как в результате я получил представление о таком принципе объектно-ориентированного программирования, как инкапсуляция.

Исходный код

main.cpp

```
#include <iostream>
#include <stdio.h>
using namespace std;
class Complex
{
public:
        double re, im;
    Complex(){}
    Complex(int x, int y)
    {
        re = x;
        im = y;
    }
        Complex add(Complex a)
        {
                    double resre, resim;
                    resre = a.re + re;
                    resim = a.im + im;
                    return Complex(resre, resim);
        }
        Complex sub(Complex a)
        {
                    double resre, resim;
                    resre = re - a.re;
                    resim = im - a.im;
                    return Complex(resre, resim);
        }
        Complex mul(Complex a)
        {
                    double resre, resim;
                    resre = a.re * re - a.im * im;
                    resim = re * a.im + im * a.re;
                    return Complex(resre, resim);
        }
        Complex div(Complex a)
        {
                    double resre, resim;
                    resre = (re * a.re + im * a.im) / (a.re*a.re + a.im*a.im);
                    resim = (a.re * im - re * a.im) / (a.re*a.re + a.im*a.im);
```

```
                return Complex(resre, resim);
        }
        bool equ(Complex a)
        {
                return (re == a.re && im == a.im);
        }
        Complex conj()
        {
                return Complex(re, -im);
        }
        ~Complex(){}
};

int main()
{
        double re, im;
        bool f;
        Complex com1, com2, com;
        printf("Enter complex number one\n");
        scanf("%lf %lf", &re, &im);
        com1 = Complex(re, im);
        printf("Enter complex number two\n");
        scanf("%lf %lf", &re, &im);
        com2 = Complex(re, im);
        com = com1.add(com2);
        printf("Sum = (%lf, %lf)\n", com.re, com.im);
        com = com1.sub(com2);
        printf("Sub = (%lf, %lf)\n", com.re, com.im);
        com = com1.mul(com2);
        printf("Mul = (%lf, %lf)\n", com.re, com.im);
        com = com1.div(com2);
        printf("Div = (%lf, %lf)\n", com.re, com.im);
        f = com1.equ(com2);
        printf("Equ = %d\n", f);
        com = com1.conj();
        printf("Conj1 = (%lf, %lf)\n", com.re, com.im);
        com = com2.conj();
        printf("Conj2 = (%lf, %lf)\n", com.re, com.im);

}
```