МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСОЙ ФЕДЕРАЦИИ МОСКОВСКИЙ АВЦИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЬЕЛЬСКИЙ УНИВЕРСТИТЕТ)

ЛАБОРАТОРНАЯ РАБОТА №1 **по курсу**

объектно-ориентированное программирование I семестр, 2019/20 уч. год

Студент *Артемьев Дмитрий Иванович, группа М8О-206Б-18*

Преподаватель *Журавлёв Андрей Андреевич*

Условие

Задание №1: написать класс, который реализует комплексные числа в алгебраической форме с операциями:

1. сложения add, (a, b) + (c, d) = (a + c, b + d);
2. вычитания sub, (a, b) – (c, d) = (a – c, b – d);
3. умножения mul, (a, b) * (c, d) = (ac – bd, ad + bc);
4. деления div, (a, b) / (c, d) = (ac + bd, bc – ad) / (c2 + d2);
5. сравнение equ, (a, b) = (c, d), если (a = c) и (b = d);
6. сопряженное число conj, conj(a, b) = (a, –b);
7. сравнения модулей.

Описание программы

Исходный код лежит в 3 файлах:

1. src/main.cpp: основная программа, которая считывает 2 комплексных числа и обрабатывает их
2. include/Complex.hpp: описание класса, объявление и реализация методов операций
3. src/Complex.cpp: реализация нереализованных методов класса комплексных чисел, чтобы было

Дневник отладки

Самое нудное в лабораторных - писать тесты и отчёты.

Недочёты

В TeXe неудобно писать, нужно допилить шаблоны для org-mode. Чекер разбит на 3 файла.

Выводы

Научился работать с git, cmake, изучил особенности работы с классами в языке C++.

Исходный код

Листинг 1: main.cpp

```cpp
#include <iostream>
#include <iomanip>
#include <cmath>
#include <Complex.hpp>

int main(){
    std::cout << std::fixed << std::setprecision(3);
    Complex a, b;
    a.read(std::cin);
    b.read(std::cin);

    std::cout << "Addition: ";
    a.add(b).write(std::cout);

    std::cout << "Subtraction: ";
    a.sub(b).write(std::cout);

    std::cout << "Multiplication: ";
    a.mul(b).write(std::cout);

    std::cout << "Division: ";
    a.div(b).write(std::cout);

    std::cout << "Comparsion a == b: " << a.equ(b) << std::endl;

    std::cout << "Conjugate numbers: \n";
    a.conj().write(std::cout);
    b.conj().write(std::cout);

    std::cout << "Module comparsion: " << a.cmp(b) << std::endl;

    return 0;
}
```

Листинг 2: Complex.hpp

```cpp
#pragma once

#include <cmath>
#include <iostream>

class Complex{
private:
    double a, b;

public:
    Complex() {};

    Complex(double a, double b):
        a(a), b(b) {};

    Complex(Complex const& other){
        this->a = other.a;
        this->b = other.b;
    }

    ~Complex (){};

    Complex add(Complex const& other) const {
        return Complex(this->a + other.a, this->b + other.b);
    }

    Complex sub(Complex const& other) const {
        return Complex(this->a - other.a, this->b - other.b);
    }

    Complex mul(Complex const& other) const {
        return Complex(this->a * other.a - this->b * other.b,
                       this->a * other.b + this->b * other.a);
    }

    Complex div(Complex const& other) const {
        double den = (other.a * other.a + other.b * other.b);
        return Complex(
                       (this->a * other.a + this->b * other.b) / den,
                       (this->b * other.a - this->a * other.b) / den
                       );
    }

    Complex conj() const {
```

```
45          return Complex(this->a, -this->b);
46      }
47
48      double mdl() const {
49          return sqrt(this->a * this->a + this->b * this->b);
50      }
51
52      bool equ(Complex const& other) const {
53          return (this->a == other.a) && (this->b == other.b);
54      }
55
56      int cmp(Complex const& other) const{
57          return (this->mdl() > other.mdl()) ? 1
58              : (this->mdl() == other.mdl()) ? 0
59              : -1;
60      }
61
62      void read(std::istream& in);
63      void write(std::ostream& out) const;
64  };
```

Листинг 3: Complex.cpp

```
1
2 #include <Complex.hpp>
3 #include <iostream>
4
5 void Complex::read(std::istream& in){
6     in >> a >> b;
7 }
8
9 void Complex::write(std::ostream& out) const{
10     out << a << ' ' << b << std::endl;
11 }
```