

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ
ФЕДЕРАЦИИ МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

ЛАБОРАТОРНАЯ РАБОТА №1 по курсу объектно-ориентированное программирование I семестр, 2019/20 уч. год

Студент Артемов 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 , $(a, b) / (c, d) = (ac + bd, bc - ad) / (c^2 + d^2)$;
5. сравнение eq , $(a, b) = (c, d)$, если $(a = c)$ и $(b = d)$;
6. сопряженное число conj , $\text{conj}(a, b) = (a, -b)$;
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 TeXе неудобно писать, нужно допилить шаблоны для org-mode. Чекер разбит на 3 файла. Недокументированный код.

Выводы

Не стоит писать лабы ночью.

Исходный код

Листинг 1: main.cpp

```
1 #include <iostream>
2 #include <iomanip>
3 #include <cmath>
4 #include <Complex.hpp>
5
6 int main(){
7     std::cout << std::fixed << std::setprecision(3);
8     Complex a, b;
9     a.read(std::cin);
10    b.read(std::cin);
11
12    std::cout << "Addition: ";
13    a.add(b).write(std::cout);
14
15    std::cout << "Subtraction: ";
16    a.sub(b).write(std::cout);
17
18    std::cout << "Multiplication: ";
19    a.mul(b).write(std::cout);
20
21    std::cout << "Division: ";
22    a.div(b).write(std::cout);
23
24    std::cout << "Comparsion a == b: " << a.equ(b) << std::endl;
25
26    std::cout << "Conjugate numbers: \n";
27    a.conj().write(std::cout);
28    b.conj().write(std::cout);
29
30    std::cout << "Module comparsion: " << a.cmp(b) << std::endl;
31
32    return 0;
33 }
```

Листинг 2: Complex.hpp

```

1 #pragma once
2
3 #include <cmath>
4 #include <iostream>
5
6 class Complex{
7 private:
8     double a, b;
9
10 public:
11     Complex() {};
12
13     Complex(double a, double b):
14         a(a), b(b) {};
15
16     Complex(Complex const& other){
17         this->a = other.a;
18         this->b = other.b;
19     }
20
21     ~Complex (){};
22
23     Complex add(Complex const& other) const {
24         return Complex(this->a + other.a, this->b + other.b);
25     }
26
27     Complex sub(Complex const& other) const {
28         return Complex(this->a - other.a, this->b - other.b);
29     }
30
31     Complex mul(Complex const& other) const {
32         return Complex(this->a * other.a - this->b * other.b,
33             this->a * other.b + this->b * other.a);
34     }
35
36     Complex div(Complex const& other) const {
37         double den = (other.a * other.a + other.b * other.b);
38         return Complex(
39             (this->a * other.a + this->b * other.b) / den,
40             (this->b * other.a - this->a * other.b) / den
41         );
42     }
43
44     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 }
```