**Ордена Трудового Красного Знамени**

Федеральное Государственное

бюджетное образовательное учреждение

высшего образования

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**МОСКОВСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**СВЯЗИ И ИНФОРМАТИКИ**

Кафедра Информационной Безопасности

**ЛАБОРАТОРНАЯ РАБОТА**

**№ 5**

по дисциплине «Объектно-ориентированное программирование система защиты информации»

на тему:

**«Понятие класса. Члены класса. Друзья класса. Перегрузка операторов»**

Вариант №16

Выполнил: студ. гр. БПЗ1501

Романюк И. Ф.

Проверил: ассисент кафедры ИБ

Барков В. В.

Москва 2018

**Цель работы**

Научиться создавать простые классы, создавать конструкторы, перегружать операции, создавать объекты классов и передавать их в функции.

**Выполнение**

Задание

Выберите для выполнения лабораторной работы свой вариант.

Для выбранного варианта определите класс, включив в него:

• конструктор по умолчанию;

• конструктор инициализации;

• конструктор преобразования базового типа к типу, определяемому разрабатываемым классом;

В разрабатываемом классе перегрузите потоковые операции для объектов класса. Для выполнения задания предложенного варианта перегрузите необходимые математические операции.

При разработке класса вашего варианта учтите индивидуальные уточнения для функций – членов класса.

Разработайте для объектов вашего класса предложенные в каждом варианте пользовательские функции.

Разработайте функцию main, организующую ввод данных и демонстрацию работы разработанных функций

Исходный код:

main.cpp

#include "Complex.h"

void main()

{

setlocale(LC\_ALL, "Rus");

double real, imaginary;

cout << "Введите \"Комплексное число\" в виде:\n-действительной\n-мнимой\nчастей комплексного числа:\nreal = ";

cin >> real;

cout << "imagine = ";

cin >> imaginary;

Complex z(real, imaginary);

cout << "z = " << z;

cout << "\nf(z) = 2z + sin(z - i) = " << Complex(2, 0)\*z << " + (" << sin(z- Complex(0, 1)) << ") = " << Complex(2, 0)\*z + sin(z - Complex(0, 1));

\_getch();

}

Complex.cpp

#include "Complex.h"

double Complex::getReal() const

{

return this->real;

}

double Complex::getImaginary() const

{

return this->imaginary;

}

void Complex::setReal(double real)

{

this->real = real;

}

void Complex::setImaginary(double imaginary)

{

this->imaginary = imaginary;

}

Complex::Complex(double real, double imaginary)

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

ostream & operator << (ostream & stream, const Complex &z)

{

if (z.getImaginary() >= 0)

{

stream << z.getReal() << "+" << z.getImaginary() << "\*i";

}

else

{

stream << z.getReal() << z.getImaginary() << "\*i";

}

return stream;

}

Complex operator -(Complex a, Complex b)

{

return Complex(a.getReal() - b.getReal(), a.getImaginary() - b.getImaginary());

}

Complex operator +(Complex a, Complex b)

{

return Complex(a.getReal() + b.getReal(), a.getImaginary() + b.getImaginary());

}

Complex operator \*(Complex a, Complex b)

{

return Complex(((a.getReal() \* b.getReal()) - (a.getImaginary()\*b.getImaginary())), ((a.getReal() \* b.getImaginary()) + (a.getImaginary()\*b.getReal())));

}

Complex operator /(Complex a, Complex b)

{

return Complex(((a.getReal() \* b.getReal()) - (a.getImaginary()\*b.getImaginary()\*(-1))) / (pow(b.getReal(), 2) + pow(b.getImaginary(), 2)), ((a.getReal()\*b.getImaginary()\*(-1)) + (a.getImaginary()\*b.getReal())) / (pow(b.getReal(), 2) + pow(b.getImaginary(), 2)));

}

Complex sin(const Complex &z)

{

return Complex(sin(z.getReal())\*cosh(z.getImaginary()), cos(z.getReal())\*sinh(z.getImaginary()));

}

Complex.h

#pragma once

#include "iostream"

#include "conio.h"

using namespace std;

class Complex

{

double real, imaginary;

public:

double getReal() const;

double getImaginary() const;

void setReal(double real);

void setImaginary(double imaginary);

Complex(double real, double imaginary);

friend ostream & operator << (ostream & stream, const Complex &z);

friend Complex operator -(Complex a, Complex b);

friend Complex operator +(Complex a, Complex b);

friend Complex operator \*(Complex a, Complex b);

friend Complex sin(const Complex &z);

};

Результаты:

