**ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ**

Ордена трудового красного знамени федеральное государственное бюджетное образовательное учреждение высшего образования

**МОСКОВСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ СВЯЗИ И ИНФОРМАТИКИ**

КАФЕДРА РАДИООБОРУДОВАНИЯ И СХЕМОТЕХНИКИ

**Лабораторная работа № 5**

по дисциплине «Объектно-ориентированное программирование систем защиты информации»:

на тему:

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

Вариант №14

Выполнил:

Студент группы БПЗ1501

Поляев Антон

Москва, 2018

# Цель работы

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

# Результаты

## 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()\*(-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 pow(const Complex &z, int stepen)

{

Complex result(0, 0);

for (int i = 0; i <= stepen; i++)

{

switch (i % 4)

{

case 1:

result.setImaginary((factorial(stepen) / (factorial(i)\*factorial(stepen - i)))\*pow(z.getReal(), stepen - i)\*pow(z.getImaginary(), i));

break;

case 2:

result.setReal((factorial(stepen) / (factorial(i)\*factorial(stepen - i)))\*pow(z.getReal(), stepen - i)\*pow(z.getImaginary(), i));

break;

case 3:

result.setImaginary((factorial(stepen) / (factorial(i)\*factorial(stepen - i)))\*pow(z.getReal(), stepen - i)\*pow(z.getImaginary(), i));

break;

case 0:

result.setReal((factorial(stepen) / (factorial(i)\*factorial(stepen - i)))\*pow(z.getReal(), stepen - i)\*pow(z.getImaginary(), i));

break;

}

}

return result;

}

Complex tanh(const Complex &z)

{

return Complex(sinh(2 \* z.getReal()), sin(2 \* z.getImaginary())) / Complex(cosh(2 \* z.getReal()) + cos(2 \* z.getImaginary()), 0);

}

## 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 pow(const Complex &z, int stepen);

friend Complex tanh(const Complex &z);

};

int factorial(int x);

## main.cpp

#include "Complex.h"

void main()

{

setlocale(LC\_ALL, "Rus");

int 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) = 1 - z^5 - th(z/2)\nf(z) = 1 - (" << pow(z, 5) << ") - (" << tanh(z / Complex(2, 0)) << ") = " << (Complex(1, 0) - pow(z, 5) - tanh(z / Complex(2, 0)));

\_getch();

}

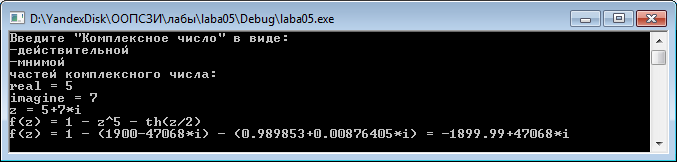
int factorial(int x)

{

if (x == 0) return 1;

return x \* factorial(x - 1);

}



1. Результат работы программы