Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации

Ордена Трудового Красного Знамени федеральное государственное

бюджетное образовательное учреждение

высшего образования

«Московский технический университет связи и информатики»

Кафедра «Информационная безопасность»

Лабораторная №4

Выполнила:

студентка группы БП31801

Зелик М.Б.

Вариант №10

Москва, 2021

**Цель работы:** научиться создавать простые классы, создавать конструкторы, перегружать операции, создавать объекты классов и передавать их в функции.

**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.

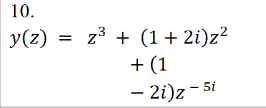
При разработке класса вашего варианта учтите индивидуальные уточнения для функций – членов класса.

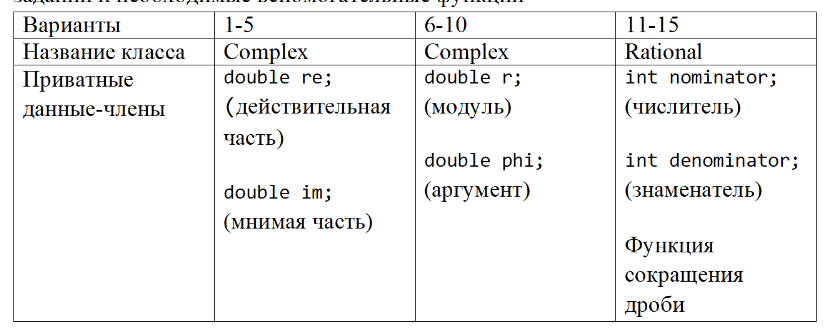
Разработайте для объектов вашего класса предложенные в каждом варианте пользовательские функции.

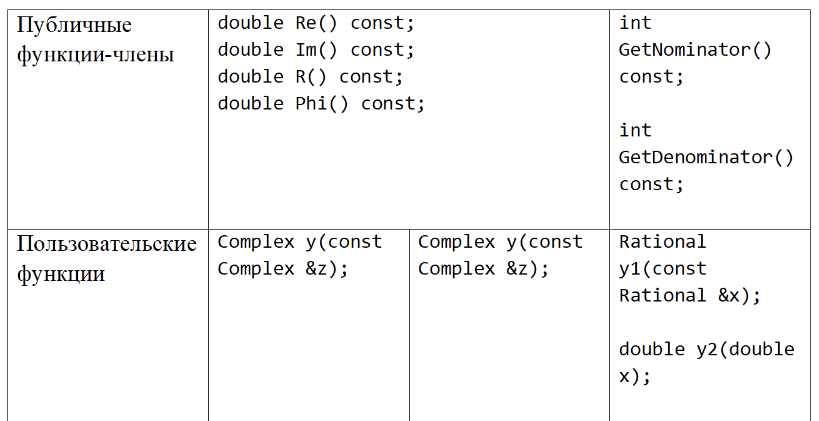
Разработайте функцию main, организующую ввод данных и демонстрацию работы разработанных функций

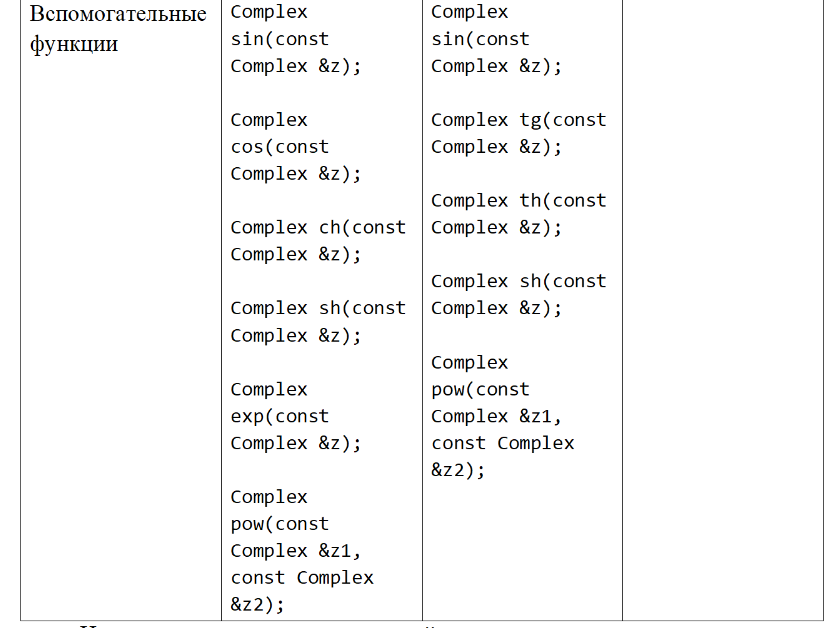
**2. Индивидуальное задание**

Вариант №10









**3. Выполнение**

#include <iostream>

#include <cmath>

#define PI 3.14159265358979323846

class Complex

{

  double r;

  double phi;

public:

  Complex() : Complex(0.0, 0.0) {}

  Complex(double x, double y) : r(x), phi(y) {}

  // Complex(double a) : Complex(a, a) {}

  double R() const { return r; }

  double Phi() const { return phi; }

  double Re() const { return cos(phi); }

  double Im() const { return sin(phi); }

  Complex operator+ (const Complex &c)

  {

    double new\_r = r + c.R();

    double new\_phi = phi + c.Phi();

    return Complex(new\_r, new\_phi);

  }

  Complex operator- (const Complex &c)

  {

    double new\_r = r - c.R();

    double new\_phi = phi - c.Phi();

    return Complex(new\_r, new\_phi);

  }

  friend Complex operator\* (const Complex &c1, const Complex &c2);

  friend Complex operator/ (const Complex &c1, const Complex &c2);

  friend Complex Pow(const Complex &z1, const Complex &z2);

  friend Complex Pow(const Complex &z, double n);

  friend std::ostream &operator<< (std::ostream &out, const Complex &c);

};

Complex operator\* (const Complex &c1, const Complex &c2)

{

  double r = c1.r \* c2.r;

  double phi = c1.phi + c2.phi;

  return Complex(r, phi);

}

Complex operator/ (const Complex &c1, const Complex &c2)

{

  double r = c1.r / c2.r;

  double phi = c1.phi - c2.phi;

  return Complex(r, phi);

}

Complex Pow(const Complex &z1, const Complex &z2)

{

}

Complex Pow(const Complex &z, double n)

{

  double r = pow(z.r, n);

  double phi = z.phi \* n;

  return Complex(r, phi);

}

std::ostream &operator<< (std::ostream &out, const Complex &c)

{

  out << c.r << "\*(cos(" << c.phi << ") + i\*sin(" << c.phi << ")" << std::endl;

  return out;

}

Complex y(const Complex &z)

{

  return Pow(z, 3) + Complex(sqrt(5), atan(2))\*Pow(z, 2) + Complex(sqrt(5), (-1\*atan(2)+2\*PI)) \* z;

}

int main()

{

  Complex z1(1, 5);

  Complex z2(0.5, 2);

  std::cout << "Complex 1: " << z1

            << "\nComplex 2: " << z2

            << std::endl;

  Complex z3 = y(z1);

  std::cout << "y(z1) = " << z3 << std::endl;

  system("pause");

  return 0;

}

Листинг 1. Программный код lab4.cpp

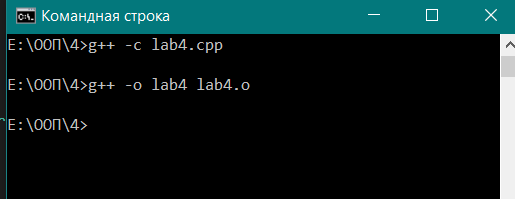


Рис 1. Сборка lab4

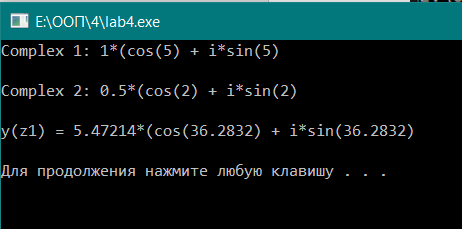


Рис 2. Результат выполнения lab4