

1. Задание

8. Разработать класс «Комплексное число», в котором данные хранятся в двух видах:

алгебраической и тригонометрической формах. Определить в нем конструкторы и деструктор, перегрузить арифметические операции, операции сравнения, операцию преобразования в строку и метод получения комплексного числа из строки, написать методы преобразования числа из одной формы в другую. Протестировать все возможности класса.

2. Листинг программы

main.cpp

```
#include "Complex.h"
```

```
int main()
```

```
{
```

```
    setlocale(LC_ALL, "Russian");
```

```
    /*ввод комплексного числа
```

```
    Complex a;
```

```
    cout << "Пожалуйста введите действительную и мнимую части " << endl;
```

```
    a.EnterCompl();
```

```
    a.PrintCopml();
```

```
    */
```

```
    /*перевод из комплексного числа в строку
```

```
    cout << string(a);
```

```
    a.PrintCopml();
```

```
    */
```

```
    /* перевод из строки в комплексное число
```

```
    Complex a("1+2i");
```

```
    cout << "Перевод строки 1+2i в комплексное число " << endl;
```

```
    cout << a;*/
```

```
    /*Перевод из показательной формы в алг.
```

```
    cout << "Перевод из показательной формы в алг. " << endl;
```

```
    Complex a;
```

```
    a.EnterCompl();
```

```
    a.Exponential();
```

```
    a.Alg();
```

```
    */
```

```
    /*сравнение комплексных чисел
```

```
    Complex a, b;
```

```
    a.EnterCompl();
```

```
    b.EnterCompl();
```

```
    a.PrintCopml();
```

```
    b.PrintCopml();
```

```
    if (a == b)
```

```
    {
```

```
        cout << "(a == b) Ровны" << endl;
```

```
    }
```

```
    else
```

```
    {
```

```
        cout << "(a == b) Ложь" << endl;
```

```
    }
```

```
    if (a != b)
```

```

    {
        cout << "(a != b) Не равны" << endl;
    }
    else
    {
        cout << "(a != b) Ложь" << endl;
    }
    if (a <= b)
    {
        cout << "(a <= b) Истина" << endl;
    }
    else
    {
        cout << "(a <= b) Ложь" << endl;
    }
    if (a >= b)
    {
        cout << "(a >= b) Истина" << endl;
    }
    else
    {
        cout << "(a >= b) Ложь" << endl;
    }
    if (a < b)
    {
        cout << "(a < b) Меньше" << endl;
    }
    else
    {
        cout << "(a < b) Ложь" << endl;
    }
    if (a > b)
    {
        cout << "(a > b) Больше" << endl;
    }
    else
    {
        cout << "(a > b) Ложь" << endl;
    }
}*/

/*арифметические операции
Complex a, b;
a.EnterCompl();
b.EnterCompl();
a.PrintCopml();
b.PrintCopml();

cout << "a + b = " << a + b;
cout << "a - b = " << a - b;
cout << "a / b = " << a / b;
cout << "a * b =" << a * b;*/

system("pause");
return 0;
}

```

Complex.h

```
#include <iostream>
```

```

#include <cmath>
#include<sstream>

const double M_PI = 3.141592653589793; // объявление константы

using namespace std;

class Complex
{
private:
    double x = 0, y = 0, fi = 0 ,r = 0;

public:
    // конструкторы
    Complex();
    Complex(double r)
    {
        x = r;
        y = 0;
    }
    Complex(double _x, double _y)
    {
        x = _x;
        y = _y;
    }
    Complex(string str)
    {
        stringstream iss(str);
        double _x, _y;
        char op;

        iss >> _x >> op >> _y;

        x = _x;
        y = _y;

        if (op == '-') _y = -_y;
    }

    //деструктор
    ~Complex();

    void EnterCompl(); // ввод комплексного числа
    void PrintCompl(); // печать комплексного числа
    void Exponential(); // перевод из алг формы в показ.
    void Alg(); // перевод из показ в алг форму

    operator string() const;

    // перегрузка операторов
    friend bool operator == (const Complex& other1, const Complex& other2);
    friend bool operator != (const Complex& other1, const Complex& other2);
    friend bool operator <= (const Complex& other1, const Complex& other2);
    friend bool operator >= (const Complex& other1, const Complex& other2);
    friend bool operator > (const Complex& other1, const Complex& other2);
    friend bool operator < (const Complex& other1, const Complex& other2);
    friend ostream& operator << (ostream& os, const Complex& other);

    //арифметические операции
    Complex operator+(Complex& b); // перегрузка сложения
    Complex operator-(Complex& b); // перегрузка вычитания
    Complex operator*(Complex& b); //перегрузка умножения
    Complex operator/(Complex& b); //перегрузка деления
};

```

Complex.cpp

```
#pragma once

#include "Complex.h"

void Complex::PrintCopml()
{
    if (y >= 0)
    {
        cout << x << "+" << y << "i\n";
    }
    else
    {
        cout << x << "-" << y << "i\n";
    }
}

void Complex::EnterCompl()
{
    cin >> x;
    cin >> y;
}

void Complex::Exponential()
{
    fi = (atan(y / x) * (180 / M_PI)); // в градусах
    r = sqrt(x * x + y * y);
    cout << r << "*" << "e^i" << fi << endl;
}

void Complex::Alg()
{
    fi = (atan(y / x) * (180 / M_PI)); // в градусах
    r = sqrt(x * x + y * y);

    double re = 0, im = 0;

    re = r * sin(fi);
    im = r * cos(fi);

    im > 0 ? cout << re << "+" << im << "i" << endl : cout << re << "-" << im << "i" <<
endl;
}

Complex::operator string() const
{
    string str = to_string(x);
    if (y >= 0)
    {
        str += " + ";
    }
    str += to_string(y) += "i\n";
    return str;
}

Complex Complex::operator+(Complex& b)
{
    Complex temp;
    temp.x = x + b.x;
    temp.y = y + b.y;

    return temp;
}
```

```

}

Complex Complex::operator-(Complex& b)
{
    Complex temp;
    temp.x = x - b.x;
    temp.y = y - b.y;

    return temp;
}

Complex Complex::operator*(Complex& b)
{
    Complex temp;
    temp.x = (x * b.x) - (y * b.y) ;
    temp.y = (x * b.y) + (b.x * y);

    return temp;
}

Complex Complex::operator/(Complex& b)
{
    Complex temp;
    temp.x = (x * b.x + y * b.y) / (b.x * b.x + b.y * b.y);
    temp.y = (b.x * y - x * b.y) / (b.x * b.x + b.y * b.y);

    return temp;
}

ostream& operator << (ostream& os, const Complex& other)
{
    if (other.y < 0)
    {
        os << other.x << "-" << other.y << "i\n";
    }
    else
    {
        os << other.x << "+" << other.y << "i\n";
    }
    return os;
}

bool operator == (const Complex& other1, const Complex& other2)
{
    return (other1.x == other2.x) && (other1.y == other2.y);
}

bool operator != (const Complex& other1, const Complex& other2)
{
    return (other1.x != other2.x) && (other1.y != other2.y);
}

bool operator <= (const Complex& other1, const Complex& other2)
{
    return (other1.x <= other2.x) && (other1.y <= other2.y);
}

bool operator >= (const Complex& other1, const Complex& other2)
{
    return (other1.x >= other2.x) && (other1.y >= other2.y);
}

bool operator > (const Complex& other1, const Complex& other2)
{
    return (other1.x > other2.x) && (other1.y > other2.y);
}

```

```
}

bool operator < (const Complex& other1, const Complex& other2)
{
    return (other1.x < other2.x) && (other1.y < other2.y);
}

Complex::Complex()
{
}

Complex::~~Complex()
{
}
```

3. Приложение

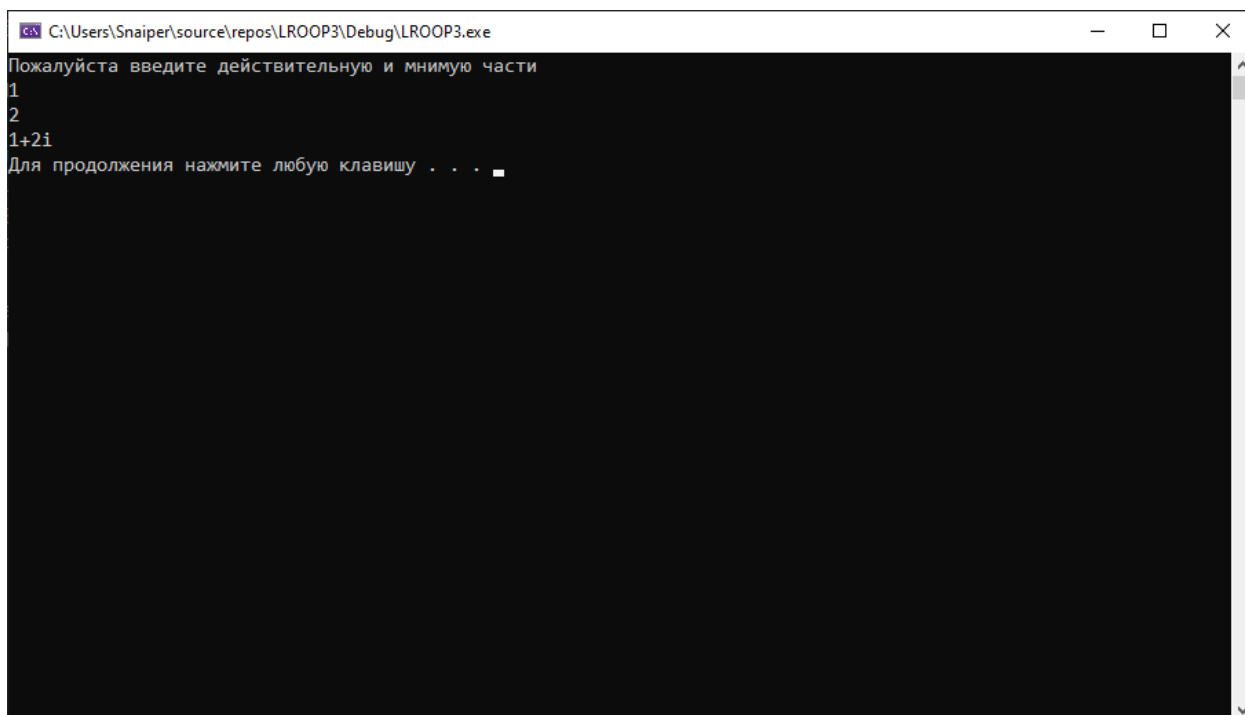


Рисунок 1 – Ввод и вывод комплексного числа в алг. форме

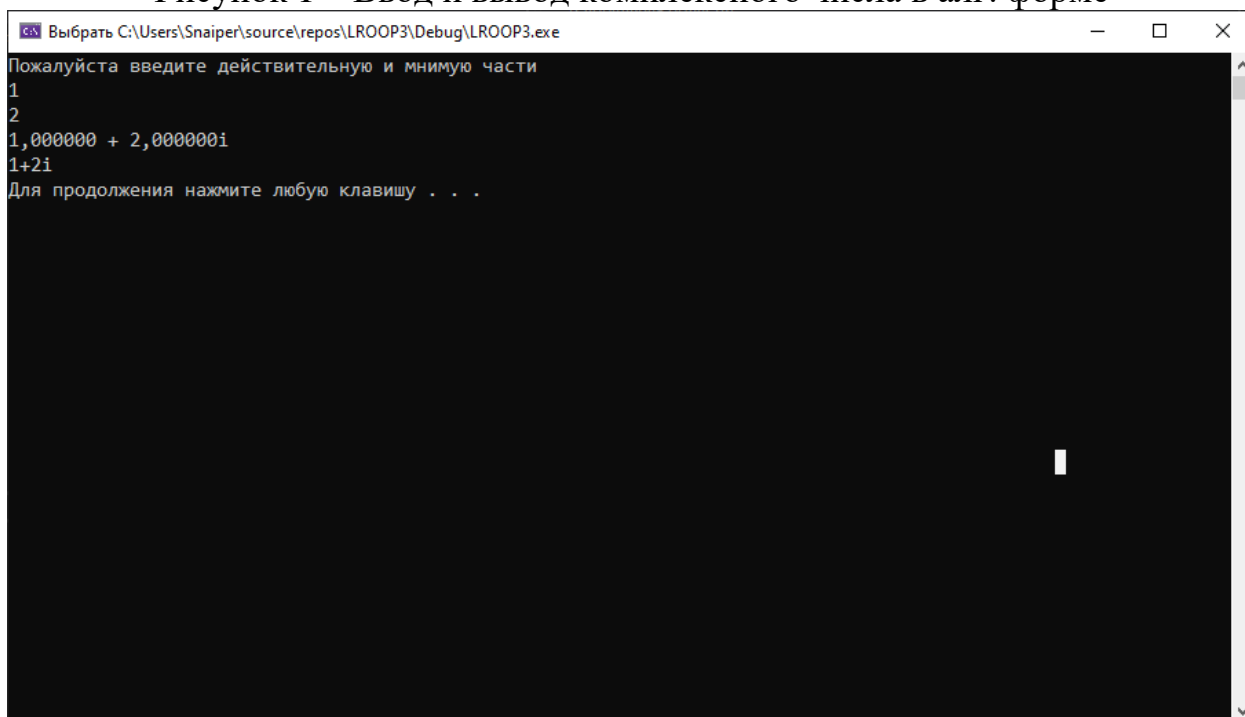


Рисунок 2 – Преобразование комплексного числа в показ. Форму

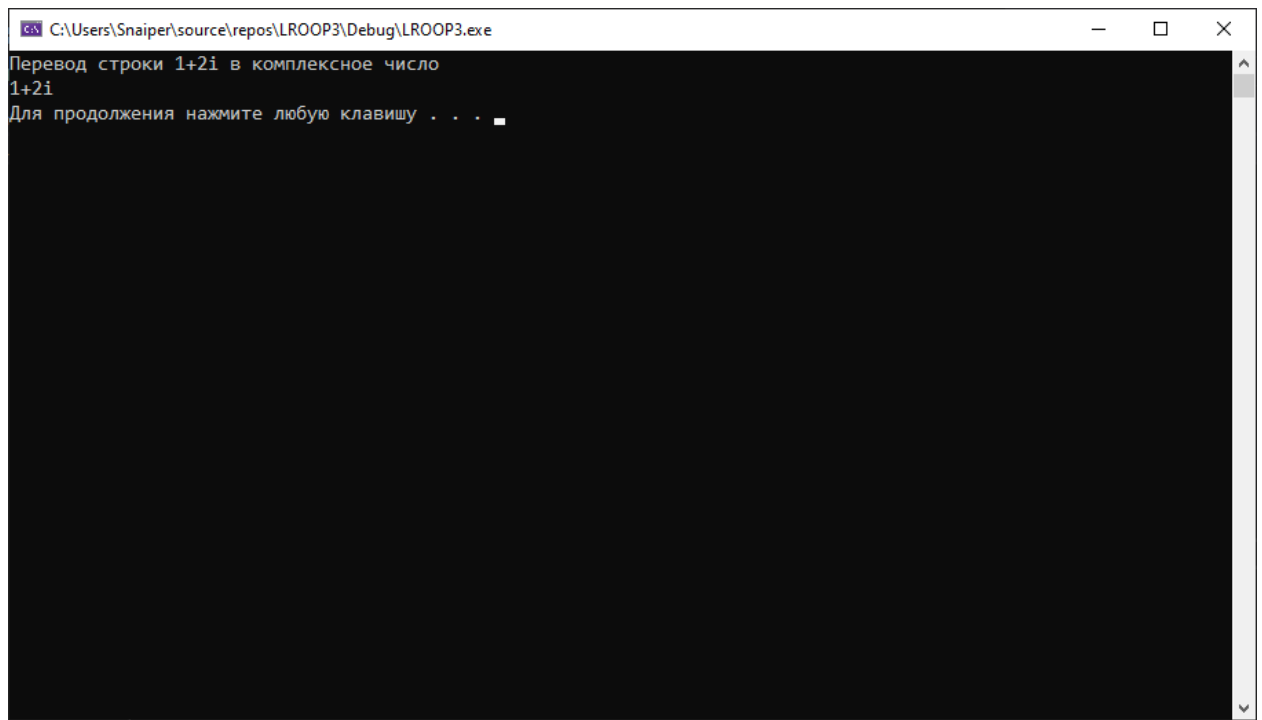


Рисунок 3 – преобразование строки в комплексное число

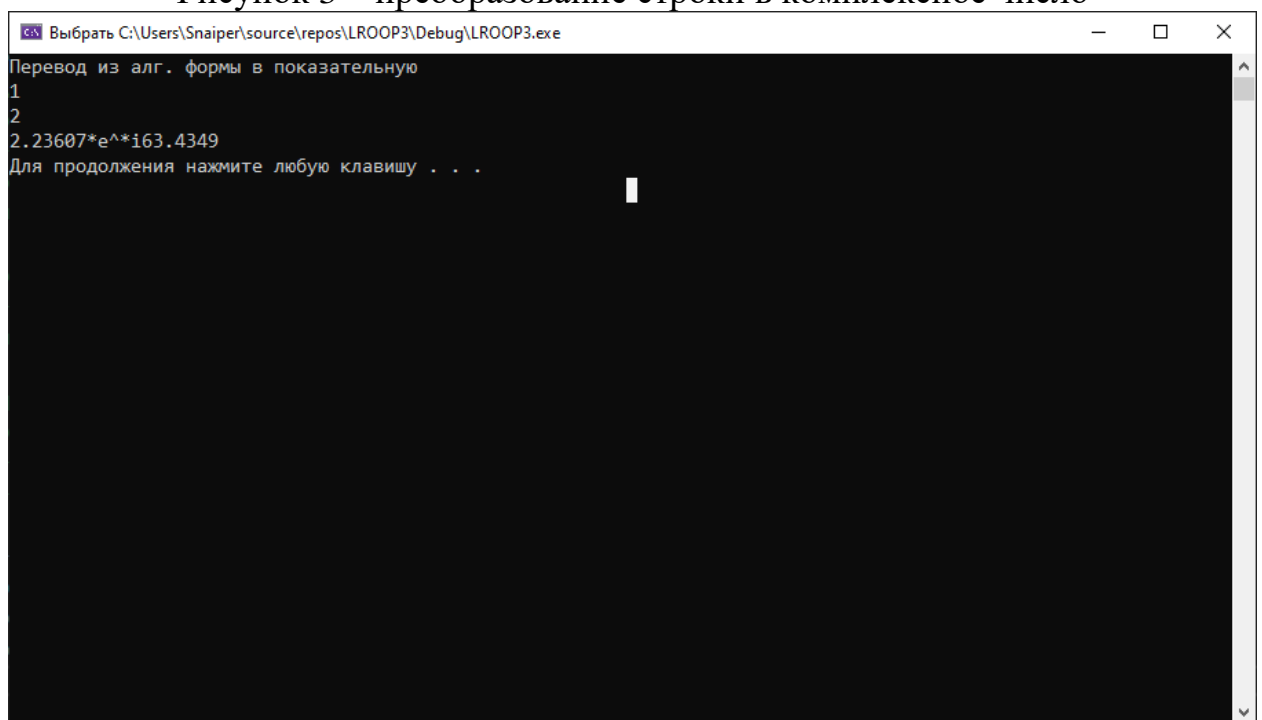


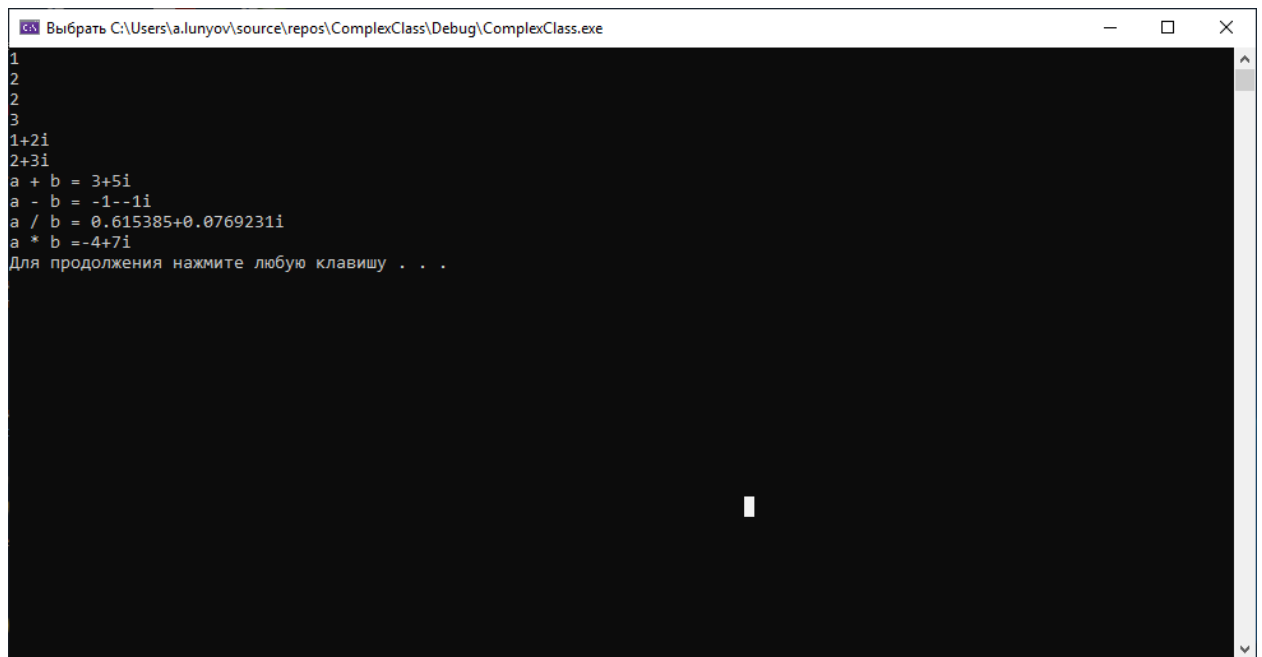
Рисунок 4 – преобразование комплексного числа в показательную форму


```
C:\Users\Snaiper\source\repos\LR00P3\Debug\LR00P3.exe
Перевод из показательной формы в алг.
1
2
2.23607*e^*i63.4349
1.26829+1.84159i
Для продолжения нажмите любую клавишу . . .
```

Рисунок 5 – преобразование комплексного числа в алг. форму

```
Выбрать C:\Users\A.lunyov\source\repos\ComplexClass\Debug\ComplexClass.exe
1
2
1
3
1+2i
1+3i
(a == b) Ложь
(a != b) Ложь
(a <= b) Истина
(a >= b) Ложь
(a < b) Ложь
(a > b) Ложь
Для продолжения нажмите любую клавишу . . .
```

Рисунок 6 – сравнение комплексных чисел



Выбрать C:\Users\A.Iunyov\source\repos\ComplexClass\Debug\ComplexClass.exe

```
1
2
2
3
1+2i
2+3i
a + b = 3+5i
a - b = -1--1i
a / b = 0.615385+0.0769231i
a * b = -4+7i
Для продолжения нажмите любую клавишу . . .
```

Рисунок 7 – арифметические операции