**Міністерство Освіти І НАУКИ України**

**Національний університет "Львівська політехніка"**

Інститут **ІКНІ**

Кафедра **ПЗ**



### ЗВІТ

До лабораторної роботи № 5

**На тему:** *“* *Створення та використання класів”*

**З дисципліни:** *“Об’єктно-орієнтоване програмування”*

**Лектор:**

доцент каф.ПЗ

Коротєєва Т. О.

**Виконав:**

ст. гр. ПЗ-16

Шеремета А.І.

**Прийняв:**

асист. каф. ПЗ

Дивак І.В.

« \_\_\_\_ » \_\_\_\_\_\_\_\_ 2022 р.

∑= \_\_\_\_ .

Львів – 2022

**Тема роботи:** Створення та використання класів.

**Мета** **роботи:** Навчитися створювати класи, використовувати конструктори для ініціалізації об’єктів, опанувати принципи створення функцій-членів. Навчитися використовувати різні типи доступу до полів та методів класів.

**Теоретичні відомості**

Клас — це визначений користувачем тип даних, який ми можемо використовувати в нашій програмі, і він працює як конструктор об’єктів або «план» для створення об’єктів.

Щоб створити клас, використовуйте ключове слово class.

Щоб створити об’єкт MyClass, вкажіть ім’я класу, а потім ім’я об’єкта.

Щоб отримати доступ до атрибутів класу використовуйте синтаксис точки (.) на об’єкті.

Конструктор класу – це спеціальна функція-член класу, яка виконується щоразу, коли ми створюємо нові об’єкти цього класу.

Конструктор матиме те саме ім'я, що й клас, і він взагалі не має жодного типу повернення, навіть void. Конструктори можуть бути дуже корисними для встановлення початкових значень для певних змінних-членів.

Конструктор за замовчуванням не має параметрів, але якщо вам потрібно, конструктор може мати параметри. Це допомагає призначити початкове значення об’єкту під час його створення.

**Лабораторне завдання**

1. Створити клас відповідно до варіанту.
2. При створенні класу повинен бути дотриманий принцип інкапсуляції.
3. Створити конструктор за замовчуванням та хоча б два інших конструктори для початкової ініціалізації об’єкта.
4. Створити функції члени згідно з варіантом.
5. Продемонструвати можливості класу завдяки створеному віконному застосуванню.
6. У звіті до лабораторної намалювати UML-діаграму класу, яка відповідає варіанту

Клас Complex – комплексне число. Клас повинен містити функції-члени, які реалізовують: а)Додавання б)Віднімання в)Множення г)Піднесення до n-го степеня д)Отримання кореня n-го степеня е)Задавання значень полів є)Зчитування (отримання значень полів) ж)Представлення в тригонометричній формі з)Введення комплексного числа з форми и)Виведення комплексного числа на форму.

**Хід роботи**

1. Створюю віконний проект.
2. Створюю потрібний клас.
3. Додаю потрібні компоненти на форму.
4. Додаємо потрібний код:

**Файл mainwindow.cpp**

complex c;

double complex::sumr(){

double sumr = R1+R2;

return sumr;

};

double complex::sumi(){

double sumi = Im1+Im2;

return sumi;

};

double complex::subtractionr(){

double subtractr = R1-R2;

return subtractr;

};

double complex::subtractioni(){

double subtracti = Im1-Im2;

return subtracti;

};

double complex::multiplyr(){

double multiplyr = ((R1\*R2)-(Im1\*Im2));

return multiplyr;

};

double complex::multiplyi(){

double multiplyi = ((R1\*Im2)+(R2\*Im1));

return multiplyi;

};

double complex::divisionr(){

double diisionr = (((R1\*R2)+(Im1\*Im2))/((R2\*R2)+(Im2\*Im2)));

return diisionr;

};

double complex::divisioni(){

double divisioni = (((Im1\*R2)-(R1\*Im2))/((R2\*R2)+(Im2\*Im2)));

return divisioni;

};

double factorial(int n) {

double result = 1;

for (int i = 1;i <= n;i++) {

result \*= i;

}

return result;

};

double complex::npowerr(int n){

double Res = 0;

for (int i = 0;i <= n;i++) {

double value = (factorial(n) / (factorial(i) \* factorial(n - i))) \* qPow(R1, n)\* qPow(Im1, i);

if (i % 2 == 0) {

Res += value;

}

}

return Res;

};

double complex::npoweri(int n){

double Res = 0;

for (int i = 0;i <= n;i++) {

double value = (factorial(n) / (factorial(i) \* factorial(n - i))) \* qPow(R1, n)\* qPow(Im1, i);

if (i % 2 != 0) {

Res += value;

}

}

return Res;

};

double complex::nrooti(int n){

double z = qPow((sqrt(R1\*R1)+(Im1\*Im1)), 1/n);

double f = (((atan(Im1/R1))+2\*M\_PI)/n);

double i = (z\*sin(f));

return i;

}

double complex::nrootr(int n){

double z = qPow((sqrt(R1\*R1)+(Im1\*Im1)), 1/n);

double f = (((atan(Im1/R1))+2\*M\_PI)/n);

double r = (z\*cos(f));

return r;

}

**Файл mainwindow.h**

class complex

{

private:

double R1,Im1,R2,Im2;

public:

complex(){

R1 = 0;

Im1 = 0;

R2 = 0;

Im2 = 0;

};

double sumi();

double sumr();

double subtractioni();

double subtractionr();

double multiplyi();

double multiplyr();

double divisioni();

double divisionr();

double npowerr(int n);

double npoweri(int n);

double nrootr(int n);

double nrooti(int n);

void setnumbers();

void setnumbers(double Real, double Imaginary, double Real2, double Imaginary2)

{

R1 = Real;

Im1 = Imaginary;

R2 = Real2;

Im2 = Imaginary2;

};

};

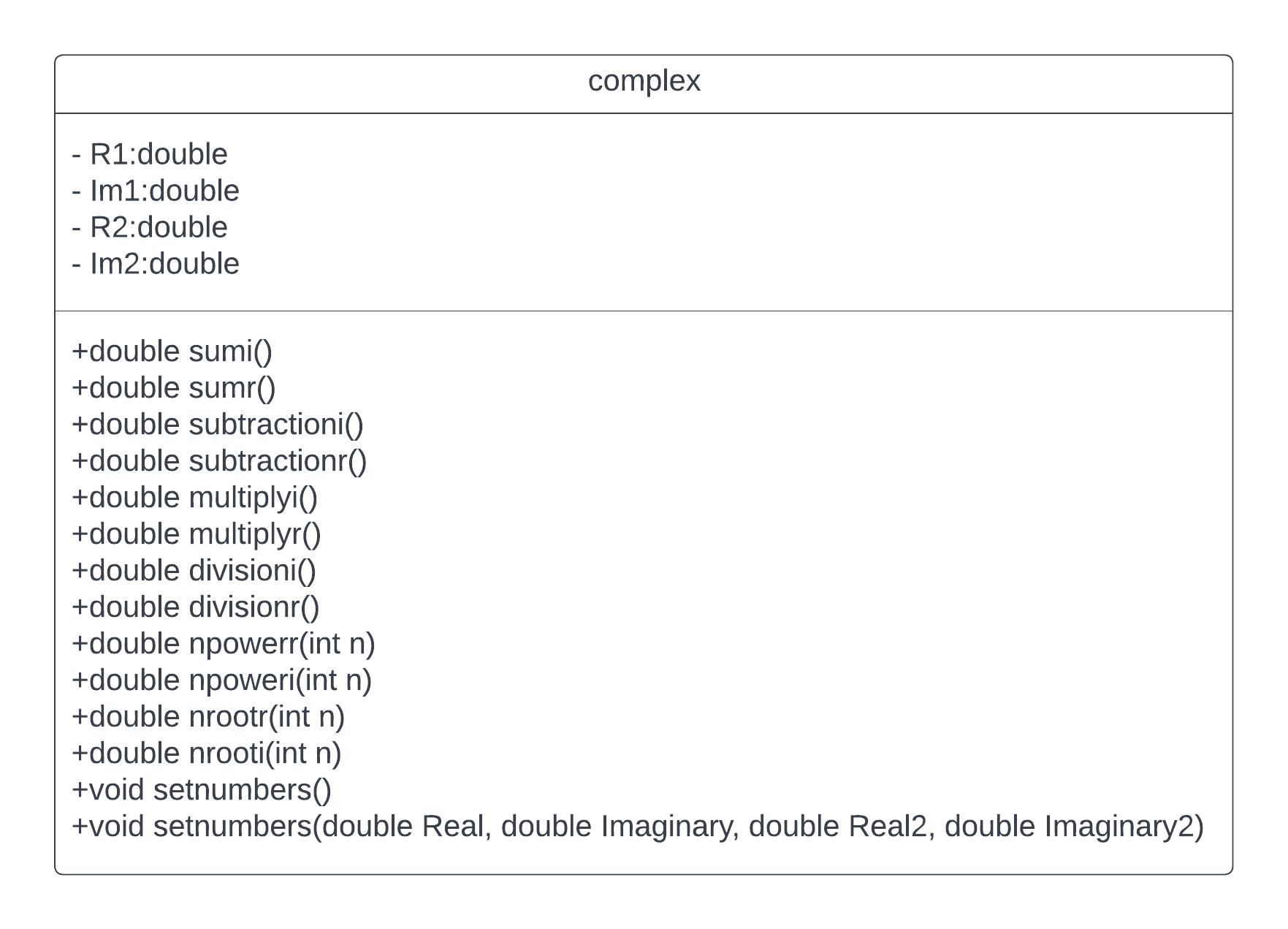


Рис. 1 UML-діаграма класу

**Виконання лаборатоної роботи**

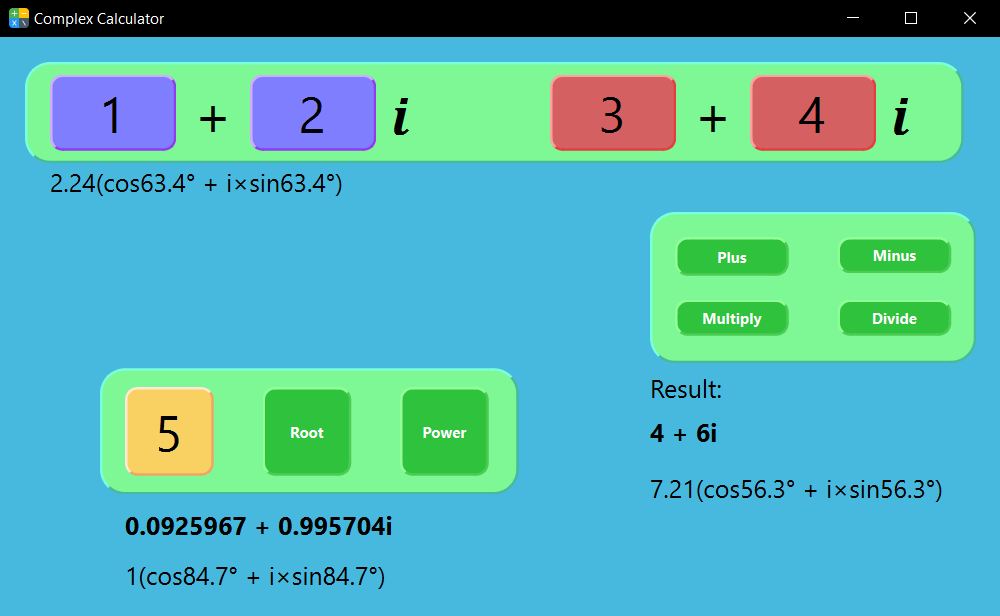


Рис. 2 Виконання програми

**Висновки**

На даній лабораторній роботі навчився створювати класи, використовувати конструктори, навчився використовувати різні типи доступу до полів та методів класів.