МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА КОМП’ЮТЕРНИХ ТЕХНОЛОГІЙ

**ЗВІТ**

лабораторної роботи №3

з дисципліни «Об’єктно орієнтоване програмування»

на тему «Створення власного класу. Методи розв’язку рівнянь»

**ВИКОНАВ:**

Студент 1-го курсу

Групи ФЕП-14

Іванський Остап Юрійович

**ПЕРЕВІРИВ:**

Асистент кафедри РКТ

Левуш Павло Назарович

Львів 2025

**МЕТА**: Засвоїти структуру опису класу у мові С++, навчитись реалізувати функціонал та використовувати класи у основній програмі.

**ПОСИЛАННЯ**: https://github.com/Ostap-Ivanskyi/OOP/tree/main/Lab4

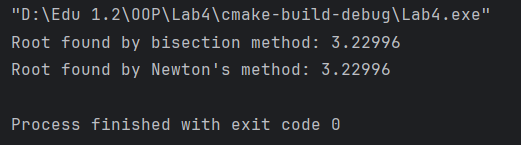
**ЗАВДАННЯ:**

Створити власні класи з реалізаціями розв’язання нелінійного рівняння методами дихотомії та дотичних з використанням принципів інкапсуляції.

Варіант №6 -функція 3x –4ln(x) – 5, проміжок - [2;4]

**ХІД РОБОТИ**

1. Створюємо заголовковий файл з описом класу Dyhotomia\_class.h та файл реалізації Dyhotomia\_class.h
2. У файлі реалізації описуємо конструктор та деструктор, сетери для полів та методи обчислення нелінійних рівнянь методом дихотомії та методом Нютона.
3. void setVolumes(double vol\_a, double vol\_b);  
     
   void setTolerance(double vol\_eps);  
     
   double solveByBisection(double (\*func)(double));  
     
   double solveByNewton(double (\*func)(double), double (\*derivative)(double), double initialGuess);
4. Метод solveByBisection() перевіряє чи є розв’язок рівняння на проміжку. Поки довжина відрізку більше значення точності розраховуємо значення центру відрізку с. якщо розв’язок на проміжку ac, змінній b присвоїти значення c, інакше – присвоїти значення c змінній a.
5. double Dyhotomia\_class::solveByBisection(double (\*func)(double)) {  
    if (func(a) \* func(b) >= 0) {  
    throw std::invalid\_argument("Function has the same signs at a and b, or no root in interval.");  
    }  
     
    double c;  
    while ((b - a) >= eps) {  
    c = (a + b) / 2;  
    if (func(c) == 0.0) {  
    return c;  
    } else if (func(c) \* func(a) < 0) {  
    b = c;  
    } else {  
    a = c;  
    }  
    }  
    return (a + b) / 2;  
   }
6. Метод solveByNewton() розвязує рівняння методом Нютона. Оголошуємо цикл в якому змінній h присвоюємо результат ділення результату функції в точці на похідну в точці. Якщо абсолютне значення h <eps перериваємо цикл та повертаємо результат.
7. double Dyhotomia\_class::solveByNewton(double (\*func)(double), double (\*derivative)(double), double initialGuess) {  
    double x = initialGuess;  
    double h;  
    int counter = 0;  
    for (int i = 0; i < 100; i++) {  
    counter ++;  
    h = func(x) / derivative(x);  
    if (std::abs(h) < eps) {  
    break;  
    }  
    x = x - h;  
    }  
    std::cout << counter << std::endl;  
    return x;  
   }
8. В main.cpp реалізовуємо дві функції double func(double x) та double derivative(double x) які рахують значення функції в точці та похідну відповідно.
9. В методі main() створюємо екземпляр класу Dyhotomia\_class, присвоюємо значення полям об’єкту. Та виводимо результат роботи методів.
10. #include <iostream>  
    #include <cmath>  
    #include "Dyhotomia\_class.h"  
      
    double func(double x) {  
     return 3 \* x - 4 \* log(x) - 5;  
    }  
      
    double derivative(double x) {  
     return 3 - 4 / x;  
    }  
      
    int main() {  
     Dyhotomia\_class solver;  
      
     solver.setVolumes(2.0, 4.0);  
     solver.setTolerance(1e-6);  
      
     try {  
     double root\_bisection = solver.solveByBisection(func);  
     std::cout << "Root found by bisection method: " << root\_bisection << std::endl;  
      
     double root\_newton = solver.solveByNewton(func, derivative, 3.0);  
     std::cout << "Root found by Newton's method: " << root\_newton << std::endl;  
     } catch (const std::invalid\_argument &e) {  
     std::cerr << "Error: " << e.what() << std::endl;  
     }  
      
     return 0;  
    }
11. Результат виконання.



**ВИСНОВОК**

Під час виконання лабораторної роботи засвоїв структуру опису класу, навчився реалізовувати функціонал та використовувати класи. Також реалізував два варіанти розв’язання нелінійних рівнянь.