МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА КОМП’ЮТЕРНИХ ТЕХНОЛОГІЙ

**ЗВІТ**

лабораторної роботи №6

з дисципліни «Об’єктно орієнтоване програмування»

на тему «Лямбда вирази»

**ВИКОНАВ:**

Студент 1-го курсу

Групи ФЕП-14

Іванський Остап Юрійович

**ПЕРЕВІРИВ:**

Асистент кафедри РКТ

Левуш Павло Назарович

Львів 2025

**МЕТА**: Вивчити синтаксис та дослідити можливості лямбда виразів у мові С++ останніх редакцій.

**ПОСИЛАННЯ**: https://github.com/Ostap-Ivanskyi/OOP/tree/main/Lab7

**ЗАВДАННЯ:**

Переписати код лабораторних розв’язок квадратного рівняння та розв’язок нелінійного рівняння методом поділу відрізку навпіл з використанням, де це доцільно, лямбда виразів. Також можна використати лабораторні із реалізацію алгоритмів з інших курсів. Пояснити, як це вплине на роботу програми.

**ХІД РОБОТИ**

1. Замість вказівників на функції тепер використовуємо std::function<double(double)>, що дозволяє передавати як звичайні функції, так і лямбда-вирази.

double solveByBisection(double (\*func)(double));  
  
double solveByNewton(double (\*func)(double), double (\*derivative)(double), double initialGuess);

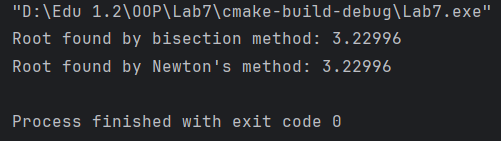
double solveByBisection(std::function<double(double)> func);  
  
double solveByNewton(std::function<double(double)> func, std::function<double(double)> derivative, double initialGuess);

1. У main.cpp функції func і derivative замінюємо на лямбда-вирази.

double root\_bisection = solver.solveByBisection([](double x) {  
 return 3 \* x - 4 \* log(x) - 5;  
});

double root\_newton = solver.solveByNewton(  
 [](double x) { return 3 \* x - 4 \* log(x) - 5; },  
 [](double x) { return 3 - 4 / x; },  
 3.0);

1. Результат виконання функцій не змінився.



**ВИСНОВОК**

1. Під час виконання лабораторної роботи ми вивчили синтаксис та дослідили можливості лямбда виразів у мові С++ останніх редакцій. Функціональність програми з використання лямбда виразів не зминалась, лямбда-вирази дозволяють інкапсулювати логіку прямо в місці її використання, що робить код більш локальним і легшим для розуміння.