Кам’янець-Подільський національний університет

 імені Івана Огієнка

КАФЕДРА КОМП’ЮТЕРНИХ НАУК

Навчальна дисципліна «Об’єктно-орієнтоване програмування»

ЛАБОРАТОРНА РОБОТА #0109

Тема: Делегати і події

Варіант №3

Виконав:

студент 1-го курсу

групи KNms1-B21

Волошин В.Ю.

Прийняла:

старший викладач,

Мястковська М.О.

Кам’янець-Подільський – 2023

1. **Хід роботи.**

Делегати та події є ключовими механізмами в мові програмування C#, що дозволяють реалізувати важливий принцип об'єктно-орієнтованого програмування - "принцип відкритості/закритості". Делегати в C# представляють собою об'єкти, які вказують на методи, і можуть бути використані для створення власних типів делегатів. Події надають механізм для підписки на події та реагування на них.

Використання делегатів та подій в C# дозволяє реалізовувати концепції реактивного програмування та створювати більш гнучкі та розширювані програми.

Делегати є типами, що визначають сигнатуру методу, до якого можна прив'язати виклик. Вони можуть бути використані для передачі методів як параметрів, що робить код більш гнучким та розширюваним. Стандартні делегати, такі як Action та Func, дозволяють передавати методи без явного визначення нового делегата.

Події в C# є механізмом, що дозволяє об'єктам надсилати повідомлення про виникнення подій та надає можливість іншим об'єктам підписуватися на ці події та реагувати на них. Це сприяє використанню моделі підписки-повідомлення в програмуванні.

Використання делегатів та подій важливо при створенні архітектури програм, де потрібно виконувати певні дії у відповідь на зміни стану або виникнення подій. Ці концепції допомагають покращити розділення відповідальностей та роблять програми більш підтримуваними та розширюваними.

1. **Завдання до виконання.**

Розробити клас для обчислення визначених інтегралів за формулою правих прямокутників з використанням делегатів.

1. **Лістинг програми.**

// Делегат для представлення функцій одного змінного, повертає значення типу double

public delegate double FunctionDelegate(double x);

// Клас для обчислення визначеного інтегралу методом правих прямокутників

public class RightRectangleIntegralCalculator

{

    // Подія, яка викликається після успішного обчислення інтегралу

    public event EventHandler<string> CalculationCompleted;

    // Метод для обчислення визначеного інтегралу методом правих прямокутників

    public double CalculateIntegral(FunctionDelegate function, double a, double b, int n)

    {

        // Визначення ширини кожного прямокутника

        double h = (b - a) / n;

        double result = 0.0;

        // Цикл для сумування значень функції на правих границях прямокутників

        for (int i = 1; i <= n; i++)

        {

            double x = a + i \* h;

            result += function(x);

        }

        // Помноження суми на ширину прямокутника

        result \*= h;

        // Виклик події для повідомлення про успішне обчислення

        OnCalculationCompleted($"Integral calculated successfully: {result}");

        return result;

    }

    // Віртуальний метод для виклику події при завершенні обчислення інтегралу

    protected virtual void OnCalculationCompleted(string message)

    {

        CalculationCompleted?.Invoke(this, message);

    }

}

class Program

{

    static void Main()

    {

        // Приклад використання класу RightRectangleIntegralCalculator

        // Функція, яку будемо інтегрувати

        FunctionDelegate function = x => x \* x;

        // Визначення меж інтегрування та кількості прямокутників

        double a = 0.0;

        double b = 1.0;

        int n = 100;

        // Створення об'єкта класу RightRectangleIntegralCalculator

        RightRectangleIntegralCalculator calculator = new RightRectangleIntegralCalculator();

        // Підписка на подію CalculationCompleted

        calculator.CalculationCompleted += (sender, message) =>

        {

            Console.WriteLine(message);

        };

        // Виклик методу для обчислення інтегралу

        double result = calculator.CalculateIntegral(function, a, b, n);

        Console.WriteLine($"Result: {result}");

    }

}

1. **Результат роботи.**



1. **Посилання на проект.**