МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНІЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Звіт

з виконання лабораторної роботи №1

«Розробка класів, що керується тестами»

з дисципліни

Програмування на С#

Виконав:

ст.гр. ІТШІ-22-1

Трущенко Тимофій

Перевірив:

Бібічков І.Є.

2023

ЛАБОРАТОРНА РОБОТА №1.

**НАЗВА РОБОТИ:** Розробка класів, що керується тестами.

**МЕТА РОБОТИ:** Вивчення особливостей розробки класів у середовищі VS 2022 з використанням підходу TDD (Test Driven Development).

**ЗАВДАННЯ:** 35. "Матриця довільної розмірності" (транспонування матриці). Дані класу: додавання та множення двох матриць, множення матриці на число, обчислення визначника матриці. Функції класу: додавання та множення двох матриць, множення матриці на число, транспонування матриці.

**ХІД РОБОТИ:**

* 1. Зображення, що містить текст, електроніка, знімок екрана, програмне забезпечення

     Автоматично згенерований описІнтерфейс класу:
  2. Розробка тестів

using Microsoft.VisualStudio.TestTools.UnitTesting;

using System;

using Trushchenko\_LW\_1;

namespace MatrixTest

{

[TestClass]

public class UnitTest1

{

[TestMethod]

public void Constructor\_WithValidArguments\_InitializesMatrix()

{

// Arrange

int rows = 3;

int columns = 4;

int[,] data = new int[3, 4] { { 1, 2, 3, 4 }, { 5, 6, 7, 8 }, { 9, 10, 11, 12 } };

// Act

Matrix matrix = new Matrix(rows, columns, data);

// Assert

Assert.IsNotNull(matrix);

Assert.AreEqual(rows, matrix.Rows);

Assert.AreEqual(columns, matrix.Columns);

// Add more assertions to verify the initialization of the matrix data.

}

[TestMethod]

public void Multiply\_ByScalar\_ReturnsCorrectResult()

{

// Arrange

int rows = 2;

int columns = 2;

int[,] data = new int[2, 2] { { 1, 2 }, { 3, 4 } };

Matrix matrix = new Matrix(rows, columns, data);

int scalar = 2;

Matrix expected = new Matrix(2, 2, new int[,] { { 2, 4 }, { 6, 8 } });

// Act

Matrix result = matrix.Multiply(scalar);

// Assert

Assert.AreEqual(expected, result);

}

[TestMethod]

public void Add\_TwoMatrices\_ReturnsCorrectResult()

{

// Arrange

int rows = 2;

int columns = 2;

int[,] data1 = new int[2, 2] { { 1, 2 }, { 3, 4 } };

int[,] data2 = new int[2, 2] { { 5, 6 }, { 7, 8 } };

Matrix matrix1 = new Matrix(rows, columns, data1);

Matrix matrix2 = new Matrix(rows, columns, data2);

Matrix expected = new Matrix(2, 2, new int[,] { { 6, 8 }, { 10, 12 } });

// Act

Matrix result = matrix1.Add(matrix2);

// Assert

Assert.AreEqual(expected, result);

}

[TestMethod]

public void Equals\_CompareMatrices\_ReturnsTrueIfEqual()

{

// Arrange

int rows = 2;

int columns = 2;

int[,] data1 = new int[2, 2] { { 1, 2 }, { 3, 4 } };

int[,] data2 = new int[2, 2] { { 1, 2 }, { 3, 4 } };

Matrix matrix1 = new Matrix(rows, columns, data1);

Matrix matrix2 = new Matrix(rows, columns, data2);

// Act

bool result = matrix1.Equals(matrix2);

// Assert

Assert.IsTrue(result);

}

[TestMethod]

public void Equals\_CompareMatrices\_ReturnsFalseIfNotEqual()

{

// Arrange

int rows = 2;

int columns = 2;

int[,] data1 = new int[2, 2] { { 1, 2 }, { 3, 4 } };

int[,] data2 = new int[2, 2] { { 1, 2 }, { 4, 3 } };

Matrix matrix1 = new Matrix(rows, columns, data1);

Matrix matrix2 = new Matrix(rows, columns, data2);

// Act

bool result = matrix1.Equals(matrix2);

// Assert

Assert.IsFalse(result);

}

[TestMethod]

public void Multiply\_TwoMatrices\_ReturnsCorrectResult()

{

// Arrange

int rows1 = 2;

int columns1 = 3;

int[,] data1 = new int[2, 3] { { 1, 2, 3 }, { 4, 5, 6 } };

Matrix matrix1 = new Matrix(rows1, columns1, data1);

int rows2 = 3;

int columns2 = 2;

int[,] data2 = new int[3, 2] { { 7, 8 }, { 9, 10 }, { 11, 12 } };

Matrix matrix2 = new Matrix(rows2, columns2, data2);

Matrix expected = new Matrix(2, 2, new int[,] { { 58, 64 }, { 139, 154 } });

// Act

Matrix result = matrix1.Multiply(matrix2);

// Assert

Assert.AreEqual(expected, result);

}

[TestMethod]

public void Transpose\_Matrix\_ReturnsTransposedMatrix()

{

// Arrange

int rows = 3;

int columns = 2;

int[,] data = new int[3, 2] { { 1, 2 }, { 3, 4 }, { 5, 6 } };

Matrix matrix = new Matrix(rows, columns, data);

Matrix expected = new Matrix(2, 3, new int[,] { { 1, 3, 5 }, { 2, 4, 6 } });

// Act

Matrix result = matrix.Transpose();

// Assert

Assert.AreEqual(expected, result);

}

}

}

* 1. Зображення, що містить текст, знімок екрана, програмне забезпечення, Мультимедійне програмне забезпечення

     Автоматично згенерований описРезультати тестів до реалізації методів класу
  2. Реалізація методів класу:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Trushchenko\_LW\_1

{

public class Matrix : IMatrix

{

private int[,] data;

public int Count { get => throw new NotImplementedException(); private set => throw new NotImplementedException(); }

private int \_rows;

public int Rows

{

get => \_rows;

set

{

if (value <= 0)

{

throw new ArgumentOutOfRangeException("The number of rows must be > 0");

}

\_rows = value;

}

}

private int \_columns;

public int Columns

{

get => \_columns;

set

{

if (value <= 0)

{

throw new ArgumentOutOfRangeException("The number of columns must be > 0");

}

\_columns = value;

}

}

public Matrix(int rows, int columns, int[,] initialData)

{

Rows = rows;

Columns = columns;

if (initialData == null)

{

throw new ArgumentNullException(nameof(initialData), "Initial data cannot be null.");

}

if (initialData.GetLength(0) != Rows || initialData.GetLength(1) != Columns)

{

throw new ArgumentException("Initial data dimensions do not match the specified matrix dimensions.");

}

data = initialData;

}

// Конструктор за розмірами матриці

public Matrix(int rows, int columns)

{

Rows = rows;

Columns = columns;

data = new int[rows, columns];

}

// Конструктор з двовимірним масивом

public Matrix(int[,] array)

{

Rows = array.GetLength(0);

Columns = array.GetLength(1);

data = new int[Rows, Columns];

for (int i = 0; i < Rows; i++)

{

for (int j = 0; j < Columns; j++)

{

data[i, j] = array[i, j];

}

}

}

// Конструктор з одновимірним масивом

public Matrix(int[] array)

{

Rows = 1;

Columns = array.Length;

data = new int[1, Columns];

for (int j = 0; j < Columns; j++)

{

data[0, j] = array[j];

}

}

// Функція порівняння об'єктів на рівність

public bool AreEqual(object obj)

{

if (!(obj is Matrix))

return false;

Matrix other = (Matrix)obj;

if (this.Rows != other.Rows || this.Columns != other.Columns)

return false;

for (int i = 0; i < Rows; i++)

{

for (int j = 0; j < Columns; j++)

{

if (this.data[i, j] != other.data[i, j])

return false;

}

}

return true;

}

public override bool Equals(object obj)

{

if (!(obj is Matrix))

return false;

Matrix other = (Matrix)obj;

if (this.Rows != other.Rows || this.Columns != other.Columns)

return false;

for (int i = 0; i < Rows; i++)

{

for (int j = 0; j < Columns; j++)

{

if (this.data[i, j] != other.data[i, j])

return false;

}

}

return true;

}

// Функція введення з консолі

public static Matrix InputFromConsole()

{

Console.Write("Введіть кількість рядків: ");

int rows = int.Parse(Console.ReadLine());

Console.Write("Введіть кількість стовпців: ");

int columns = int.Parse(Console.ReadLine());

Matrix matrix = new Matrix(rows, columns);

Console.WriteLine("Введіть елементи матриці:");

for (int i = 0; i < rows; i++)

{

for (int j = 0; j < columns; j++)

{

Console.Write($"matrix[{i}, {j}] = ");

matrix.data[i, j] = int.Parse(Console.ReadLine());

}

}

return matrix;

}

// Перевизначена функція ToString

public override string ToString()

{

string result = "";

for (int i = 0; i < Rows; i++)

{

for (int j = 0; j < Columns; j++)

{

result += data[i, j] + "\t";

}

result += "\n";

}

return result;

}

// Функція для додавання двох матриць

public Matrix Add(Matrix other)

{

if (this.Rows != other.Rows || this.Columns != other.Columns)

throw new ArgumentException("Розміри матриць не відповідають");

Matrix result = new Matrix(Rows, Columns);

for (int i = 0; i < Rows; i++)

{

for (int j = 0; j < Columns; j++)

{

result.data[i, j] = this.data[i, j] + other.data[i, j];

}

}

return result;

}

// Функція для множення матриці на число

public Matrix Multiply(int scalar)

{

Matrix result = new Matrix(Rows, Columns);

for (int i = 0; i < Rows; i++)

{

for (int j = 0; j < Columns; j++)

{

result.data[i, j] = this.data[i, j] \* scalar;

}

}

return result;

}

// Функція для множення двох матриць

public Matrix Multiply(Matrix other)

{

if (this.Columns != other.Rows)

throw new ArgumentException("Розміри матриць не відповідають для множення");

Matrix result = new Matrix(this.Rows, other.Columns);

for (int i = 0; i < this.Rows; i++)

{

for (int j = 0; j < other.Columns; j++)

{

int sum = 0;

for (int k = 0; k < this.Columns; k++)

{

sum += this.data[i, k] \* other.data[k, j];

}

result.data[i, j] = sum;

}

}

return result;

}

// Функція для транспонування матриці

public Matrix Transpose()

{

Matrix result = new Matrix(Columns, Rows);

for (int i = 0; i < Rows; i++)

{

for (int j = 0; j < Columns; j++)

{

result.data[j, i] = this.data[i, j];

}

}

return result;

}

public int this[int row, int column]

{

get

{

// Отримання значення елемента матриці за індексами row та column

return data[row, column];

}

set

{

// Встановлення значення елемента матриці за індексами row та column

data[row, column] = value;

}

}

}

}

* 1. Результати тестів після реалізації методів класу:

Зображення, що містить текст, знімок екрана, програмне забезпечення, Комп’ютерна піктограма

Автоматично згенерований опис

**Висновок:** На практичному занятті ознайомився з особливостями підходу TDD (Test Driven Development). З використанням цього підходу виконав лабораторну роботу. А саме - розробив тести до методів класу Matrix. Методи, що підлягали тестуванню: додавання, множення двох матриць,  транспонування матриці, множення матриці на число та ін. Тести проведені для неповної та повної реалізацій класу.