**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ**

**УНИВЕРСИТЕТ ИМЕНИ П.О.СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

ЛАБОРАТОРНАЯ РАБОТА №2

по дисциплине: «Объектно-ориентированное проектирование и программирование»

на тему: **«**Особенности разработки и верификации в объектно-ориентированном языке**»**

Выполнил: студент гр. ИТП-21

Кислов Я.В.

Проверил: преподаватель

Карась О.В.

Гомель 2025

**Цель работы:**

Разработать тип и выполнить для него перегрузку операций, согласно варианта. Тип должен быть размещён в библиотеке классов. Весь код должен быть снабжён элементами документирования.  
Разработать не менее 15 модульных тестов для тестирования созданного класса. Создать консольное приложение с интерфейсом пользователя, подключить библиотеку, сгенерировать XML-документацию и верифицировать приложение.

Вариант 2

**Разработать тип для работы с матрицами.** Реализовать методы, позволяющие выполнять операции умножения матриц, умножения матрицы на число, предусмотрев возможность их выполнения.

**Ход работы:**

Создаем класс MatrixType, в котором выполняем все условия задания. В моем случае это умножение матрицы на матрицу, а также умножение матрицу на число. У нас есть закрытое поле с двумерным массивом, это и будет наша матрица, также создаем конструктор, индексатор и 3 статического метода с словом operator для реализации операций с матрицами. Также были сделаны модульные тесты в MSTests. На рисунке 1 показано выполнение модульных тестов.

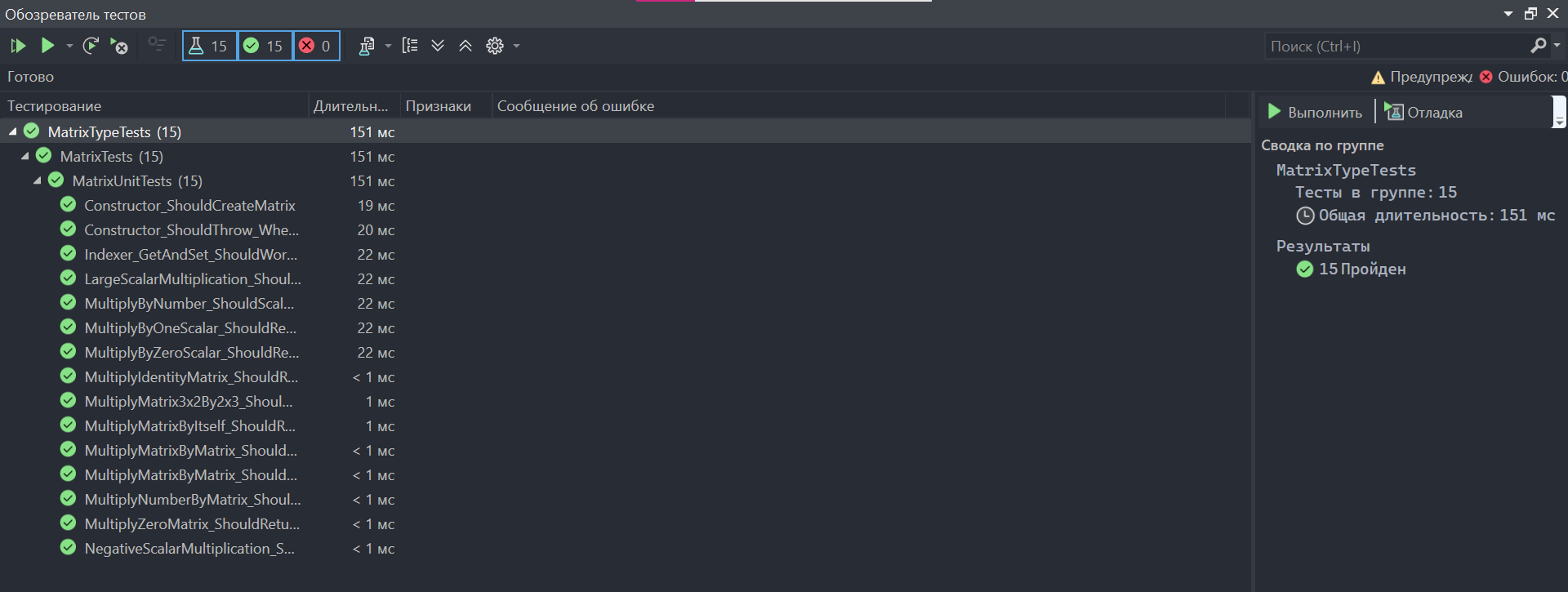


Рисунок 1 – Результат выполнения тестов

**Вывод:** В ходе работы был успешно разработан класс для работы с матрицами, реализующий операции умножения матриц и умножения матрицы на число. Библиотека классов была протестирована с помощью модульных тестов и интегрирована в консольное приложение. Результаты верификации подтвердили корректность выполнения операций и надёжность разработанного программного обеспечения.

**Приложение А**

Код программы

namespace MatrixType

{

/// <summary>

/// Представляет двумерную матрицу и поддерживает операции умножения

/// матриц и умножения матрицы на число

/// </summary>

public class Matrix

{

/// <summary>

/// Двумерный массив, в котором хранятся элементы матрицы

/// </summary>

private double[,] \_matrix;

/// <summary>

/// Количество строк в матрице

/// </summary>

private readonly int rowCount;

/// <summary>

/// Количество столбцов в матрице

/// </summary>

private readonly int colCount;

/// <summary>

/// Создаёт экземпляр матрицы на основе двумерного массива

/// </summary>

/// <param name="matrix">Двумерный массив элементов</param>

/// <exception cref="Exception">Бросается, если переданный массив равен null</exception>

public Matrix(double[,] matrix)

{

if (matrix == null)

throw new Exception("Matrix is null!");

\_matrix = matrix;

rowCount = matrix.GetUpperBound(0) + 1;

colCount = matrix.Length / rowCount;

}

/// <summary>

/// Возвращает количество столбцов в матрице

/// </summary>

public int CountCols => colCount;

/// <summary>

/// Возвращает количество строк в матрице

/// </summary>

public int CountRows => rowCount;

/// <summary>

/// Индексатор для доступа к элементам матрицы по строке и столбцу

/// </summary>

/// <param name="row">Индекс строки</param>

/// <param name="col">Индекс столбца</param>

/// <returns>Элемент матрицы в позиции [row, col]</returns>

public double this[int row, int col]

{

get => \_matrix[row, col];

set => \_matrix[row, col] = value;

}

/// <summary>

/// Умножает одну матрицу на другую

/// </summary>

/// <param name="a">Первая матрица</param>

/// <param name="b">Вторая матрица</param>

/// <returns>Новая матрица — результат умножения</returns>

/// <exception cref="Exception">Бросается, если матрицы несовместимы по размерности</exception>

public static Matrix operator \*(Matrix a, Matrix b)

{

if (a is null || b is null || a.colCount != b.rowCount)

throw new Exception("Нельзя умножить: кол-во столбцов первой матрицы не равно кол-ву строк второй.");

double[,] result = new double[a.rowCount, b.colCount];

for (int i = 0; i < a.rowCount; i++)

{

for (int j = 0; j < b.colCount; j++)

{

double sum = 0;

for (int k = 0; k < a.colCount; k++)

{

sum += a[i, k] \* b[k, j];

}

result[i, j] = sum;

}

}

return new Matrix(result);

}

/// <summary>

/// Умножает матрицу на число

/// </summary>

/// <param name="matrix">Исходная матрица</param>

/// <param name="number">Число</param>

/// <returns>Новая матрица — результат умножения</returns>

public static Matrix operator \*(Matrix matrix, double number)

{

double[,] result = new double[matrix.rowCount, matrix.colCount];

for (int i = 0; i < matrix.rowCount; i++)

{

for (int j = 0; j < matrix.colCount; j++)

{

result[i, j] = matrix[i, j] \* number;

}

}

return new Matrix(result);

}

/// <summary>

/// Умножает число на матрицу

/// </summary>

/// <param name="number">Число</param>

/// <param name="matrix">Матрица</param>

/// <returns>Новая матрица — результат умножения</returns>

public static Matrix operator \*(double number, Matrix matrix) => matrix \* number;

}

}

Приложение А1 – Код класса MatrixType

using MatrixType;

namespace MatrixApp

{

internal class Program

{

static void Main(string[] args)

{

try

{

Console.WriteLine("Для работы требуется создать матрицы.");

Console.WriteLine("Матрица А: ");

Matrix A = CreateMatrix();

Console.WriteLine("Матрица Б: ");

Matrix B = CreateMatrix();

while (true)

{

Console.WriteLine("\n--- Меню ---");

Console.WriteLine("1. Показать матрицу A");

Console.WriteLine("2. Показать матрицу Б");

Console.WriteLine("3. Умножить A на число");

Console.WriteLine("4. Умножить A на B");

Console.WriteLine("5. Выход");

Console.Write("Выберите пункт: ");

string? input = Console.ReadLine();

if (input is null)

{

throw new Exception("Ввод пуст!");

}

Console.WriteLine();

switch (input)

{

case "1":

ShowMatrix(A);

break;

case "2":

ShowMatrix(B);

break;

case "3":

Console.WriteLine("Введите число для умножения: ");

double number = double.Parse(Console.ReadLine());

ShowMatrix(A \* number);

break;

case "4":

ShowMatrix(A \* B);

break;

case "5":

return;

default:

Console.WriteLine("Неверный ввод. Введите от 1 до 5!");

break;

}

}

}

catch (Exception e)

{

Console.WriteLine($"Возникла ошибка: {e.Message}");

}

}

public static Matrix CreateMatrix()

{

Console.WriteLine("Введите кол-во строк и столбцов: ");

int rows = Int32.Parse(Console.ReadLine());

int cols = Int32.Parse(Console.ReadLine());

double[,] tempM = new double[rows, cols];

Console.WriteLine("Введите элементы матрицы: ");

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

{

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

{

tempM[i,j] = double.Parse(Console.ReadLine());

}

}

return new Matrix(tempM);

}

public static void ShowMatrix(Matrix matrix)

{

for(int i = 0;i<matrix.CountRows;i++)

{

for(int j = 0; j<matrix.CountCols;j++)

{

Console.Write(matrix[i,j] + " ");

}

Console.WriteLine();

}

}

}

}

Приложение А.2 – Код главного класса Program