

Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей
Кафедра Информатики
Дисциплина «Программирование»

ОТЧЕТ

к лабораторной работе №7

на тему:

«ПЕРЕГРУЗКА ОПЕРАТОРОВ.»

БГУИР 6-05-0612-02 17

Выполнил студент группы 353502
ХАРИТОНЧИК Денис Сергеевич

(дата, подпись студента)

Проверил ассистент каф.
Информатики
РОМАНЮК Максим Валерьевич

(дата, подпись преподавателя)

Минск 2024

1 ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

Задание 1. Вариант 7. Спроектировать класс согласно варианту индивидуального задания. Для класса использовать отдельный модуль. Спроектировать конструкторы и свойства с контролем корректности вводимых значений. Перегрузить метод `toString()`, добавить индексирование для получения полей класса. Перегрузить операции: а) математические (имеющие смысл для объектов класса), б) инкремент и декремент (изменить поля на 1), в) отношения (`==`, `!=`, `<`, `>`), г) `true` и `false`, д) преобразования типа. В методе `main()`: Создать несколько объектов класса. Продемонстрировать использование конструкторов и свойств. Продемонстрировать работу всех методов и операций. Класс квадратная матрица 2×2 типа `int`. К полям обращаться через индексатор. Перегрузить `+`, `-`, `++`, `--`, `*`, `*` на число, `/` на число. Сравнить на `==` и `!=`. (д) если определитель = 0, матрица = `false`. Преобразовать в число (опредетитель) и назад (a,0,0,a) – в обоих случаях явно.

2 ВЫПОЛНЕНИЕ РАБОТЫ

В рамках проекта был создан модуль `SquareModule`, в котором размещается один файл `SquareMatrix` с классом `SquareMatrix`. Этот класс является основным компонентом программы и демонстрации работы с матрицей.

Класс `SquareMatrix` является единственным классом в модуле. Он содержит приватное поле и публичный индексатор для доступа к элементам матрицы. Также он содержит два конструктора: один заполняет по умолчанию нулями, второй принимает параметрами для инициализации матрицы.

В классе определены все методы для перезагрузки различных операторов и метод для подсчета матрицы. Ниже приведён листинг кода класса `SquareMatrix`.

```
public class SquareMatrix
{
    private int[,] matrix = new int[2, 2];
    // индексатор для досутупа к элементам матрицы
    public int this[int row, int col]
    {
        get { return matrix[row, col]; }
```

```

        set { matrix[row, col] = value; }
    }

    // конструктор
    public SquareMatrix()
    {
        // заполняем по умолчанию 0
        for (int i = 0; i < 2; i++)
        {
            for (int j = 0; j < 2; j++)
            {
                matrix[i, j] = 0;
            }
        }
    }

    // конструктор с параметрами для инициализации матрицы
    public SquareMatrix(int a, int b, int c, int d)
    {
        matrix[0, 0] = a;
        matrix[0, 1] = b;
        matrix[1, 0] = c;
        matrix[1, 1] = d;
    }

    // перегрузка ToString
    public override string ToString()
    {
        return $"[{matrix[0, 0]}, {matrix[0, 1]}\n {matrix[1, 0]}, {matrix[1,
1]}}]";
    }

    // перегрузка сложения матрицы
    public static SquareMatrix operator +(SquareMatrix m1, SquareMatrix m2)
    {
        SquareMatrix result = new SquareMatrix();
        for (int i = 0; i < 2; i++)
        {
            for (int j = 0; j < 2; j++)
            {
                result[i, j] = m1[i, j] + m2[i, j];
            }
        }
        return result;
    }

    // перегрузка вычитания матрицы
    public static SquareMatrix operator -(SquareMatrix m1, SquareMatrix m2)
    {
        SquareMatrix result = new SquareMatrix();
        for (int i = 0; i < 2; i++)
        {
            for (int j = 0; j < 2; j++)
            {

```

```

        result[i, j] = m1[i, j] - m2[i, j];
    }
}
return result;
}

// перегрузка оператора умножения на число
public static SquareMatrix operator *(SquareMatrix m, int scalar)
{
    SquareMatrix result = new SquareMatrix();
    for (int i = 0; i < 2; i++)
    {
        for (int j = 0; j < 2; j++)
        {
            result[i, j] = m[i, j] * scalar;
        }
    }
    return result;
}

// перегрузка оператора деления
public static SquareMatrix operator /(SquareMatrix m, int divisor)
{
    if (divisor == 0)
    {
        throw new ArgumentException("На ноль делить нельзя!");
    }
    SquareMatrix result = new SquareMatrix();
    for (int i = 0; i < 2; i++)
    {
        for (int j = 0; j < 2; j++)
        {
            result[i, j] = m[i, j] / divisor;
        }
    }
    return result;
}

// перегрузка инкремента ++matrix
public static SquareMatrix operator ++(SquareMatrix m)
{
    SquareMatrix result = new SquareMatrix();
    for (int i = 0; i < 2; i++)
    {
        for (int j = 0; j < 2; j++)
        {
            result[i, j] = m[i, j] + 1;
        }
    }
    return result;
}

// перегрузка инкремента --matrix
public static SquareMatrix operator --(SquareMatrix m)

```

```

{
    SquareMatrix result = new SquareMatrix();
    for (int i = 0; i < 2; i++)
    {
        for (int j = 0; j < 2; j++)
        {
            result[i, j] = m[i, j] - 1;
        }
    }
    return result;
}

// перегрузка оператора равенства ==
public static bool operator ==(SquareMatrix m1, SquareMatrix m2)
{
    for (int i = 0; i < 2; i++)
    {
        for (int j = 0; j < 2; j++)
        {
            if (m1[i, j] != m2[i, j])
            {
                return false;
            }
        }
    }
    return true;
}

// перегрузка оператора неравенства
public static bool operator !=(SquareMatrix m1, SquareMatrix m2)
{
    return !(m1 == m2);
}

// перегрузка оператора true
public static bool operator true(SquareMatrix m)
{
    return m.Determinant() != 0;
}

// перегрузка оператора false
public static bool operator false(SquareMatrix m)
{
    return m.Determinant() == 0;
}

// преобразователь типа SquareMatrix в int ( определитель )
public static explicit operator int(SquareMatrix m)
{
    return m.Determinant();
}

private int Determinant()
{

```

```

        return matrix[0, 0] * matrix[1, 1] - matrix[0, 1] * matrix[1, 0];
    }
}

```

Класс Program представляет точку входа в приложение. Внутри метода Main создаются объекты класса SquareMatrix. Происходит вся инициализация объектов и создание 2 матриц. Затем продемонстрирована работа всех методов перегрузки. Также реализована проверка ввода данных, чтобы нельзя было создать матрицу не 2x2.

Этот класс обеспечивает демонстрацию работы модуля и работы его методов. Ниже представлен листинг кода класса Program. На рисунке 1 изображен вывод данных в консоль с корректными данными. На рисунке 2 изображен вывод в консоль с некорректными данными.

```

class Program
{
    static void Main(string[] args)
    {
        try
        {
            // Создание матриц
            SquareMatrix matrix1 = new SquareMatrix(1, 2, 3, 4);
            SquareMatrix matrix2 = new SquareMatrix(2, 4, 6, 8);

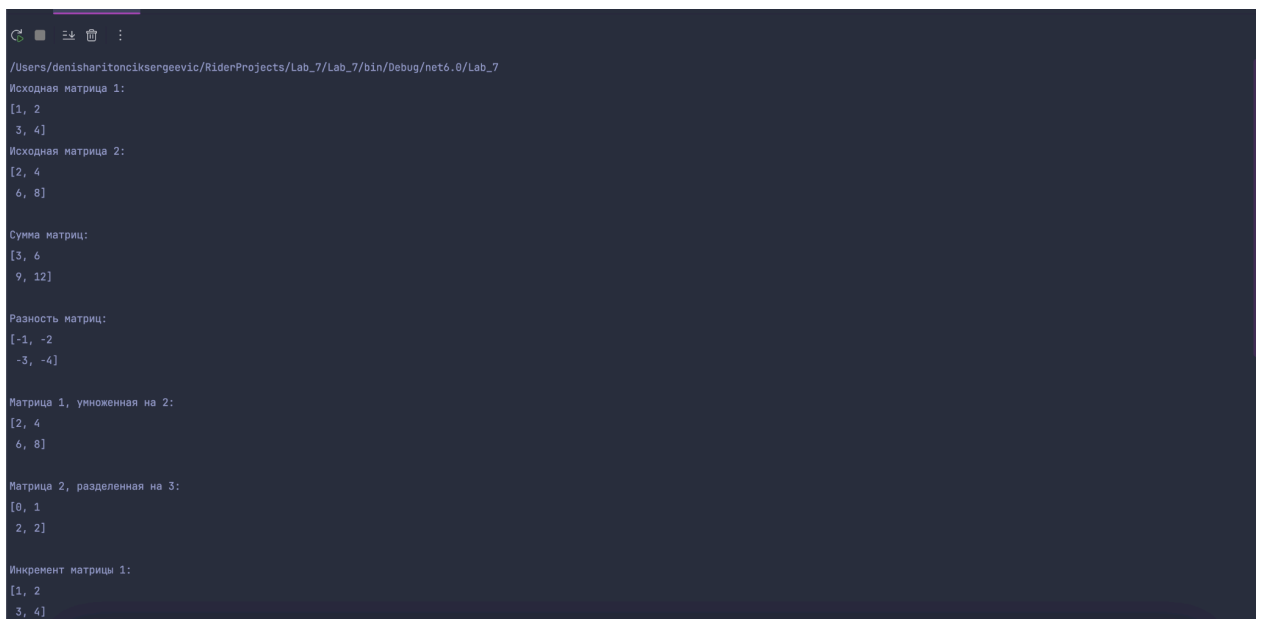
            // SquareMatrix invalidMatrix = new SquareMatrix(new int[,] { { 1,
            2, 3 }, { 4, 5, 6 }, { 7, 8, 9 } });
            // Вывод исходных матриц
            Console.WriteLine("Исходная матрица 1:");
            Console.WriteLine(matrix1.ToString());
            Console.WriteLine("Исходная матрица 2:");
            Console.WriteLine(matrix2.ToString());
            // Выполнение различных операций с матрицами
            SquareMatrix sum = matrix1 + matrix2;
            Console.WriteLine("\nСумма матриц:");
            Console.WriteLine(sum.ToString());
            SquareMatrix difference = matrix1 - matrix2;
            Console.WriteLine("\nРазность матриц:");
            Console.WriteLine(difference.ToString());
            int scalar = 2;
            SquareMatrix scaledMatrix = matrix1 * scalar;
            Console.WriteLine($"Матрица 1, умноженная на {scalar}:");
            Console.WriteLine(scaledMatrix.ToString());
            scalar = 3;
            SquareMatrix dividedMatrix = matrix2 / scalar;
            Console.WriteLine($"Матрица 2, разделенная на {scalar}:");
            Console.WriteLine(dividedMatrix.ToString());
            Console.WriteLine("\nИнкремент матрицы 1:");
            Console.WriteLine((matrix1++).ToString());
            Console.WriteLine("\nДекремент матрицы 2:");
            Console.WriteLine(--matrix2.ToString());
        }
    }
}

```

```

        Console.WriteLine("\nПроверка на равенство:");
        Console.WriteLine(matrix1 == matrix2); // должно вернуть false
        Console.WriteLine("\nПроверка на неравенство:");
        Console.WriteLine(matrix1 != matrix2); // должно вернуть true
        Console.WriteLine("\nПроверка условия true:");
        Console.WriteLine(matrix1 ? "Матрица 1 не вырожденная." : "Матрица
1 вырожденная.");
        Console.WriteLine("\nПроверка условия false:");
        Console.WriteLine(matrix2 ? "Матрица 2 не вырожденная." : "Матрица
2 вырожденная.");
        int determinant1 = (int)matrix1;
        Console.WriteLine($"Определитель матрицы 1: {determinant1}");
        SquareMatrix reconstructedMatrix = new SquareMatrix(determinant1,
0, 0, determinant1);
        Console.WriteLine("\nВосстановленная матрица из определителя:");
        Console.WriteLine(reconstructedMatrix.ToString());
    }
    catch (ArgumentException ex)
    {
        Console.WriteLine($"Ошибка: {ex.Message}");
    }
}
}

```

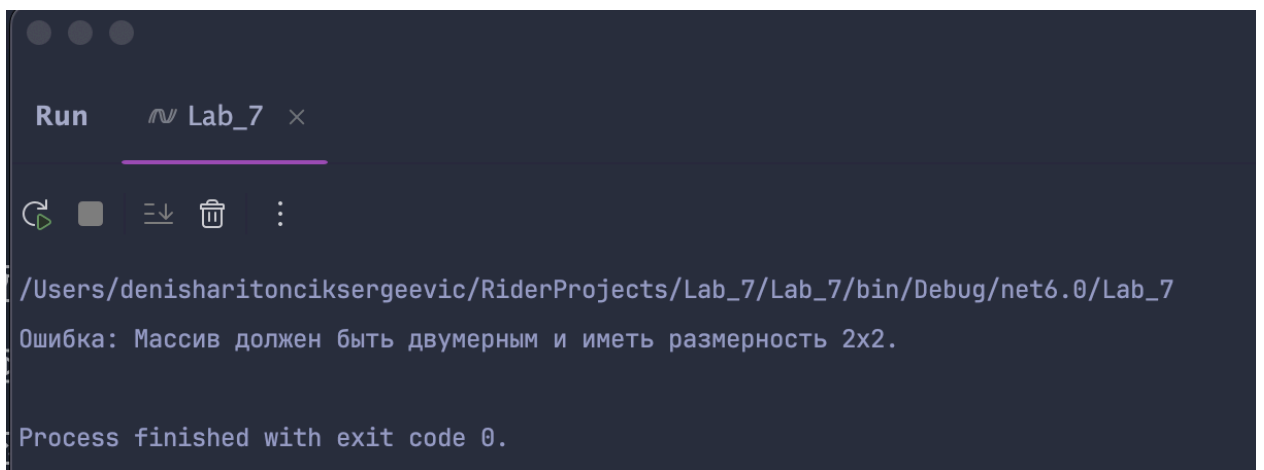


```

/Users/denisharitonciksergeevic/RiderProjects/Lab_7/Lab_7/bin/Debug/net6.0/Lab_7
Исходная матрица 1:
[1, 2
 3, 4]
Исходная матрица 2:
[2, 4
 6, 8]
Сумма матриц:
[3, 6
 9, 12]
Разность матриц:
[-1, -2
 -3, -4]
Матрица 1, умноженная на 2:
[2, 4
 6, 8]
Матрица 2, разделенная на 3:
[0, 1
 2, 2]
Инкремент матрицы 1:
[1, 2
 3, 4]

```

Рисунок 1 – Вывод в консоль с корректными данными

A screenshot of a Visual Studio console window. The title bar shows 'Run' and 'Lab_7' with a close button. Below the title bar is a toolbar with icons for running, stopping, and debugging. The console output shows the file path: `/Users/denisharitonciksergeevic/RiderProjects/Lab_7/Lab_7/bin/Debug/net6.0/Lab_7`. Below the path, there is an error message in Russian: `Ошибка: Массив должен быть двумерным и иметь размерность 2x2.`. At the bottom, it says `Process finished with exit code 0.`

```
Run  Lab_7 ×  
/Users/denisharitonciksergeevic/RiderProjects/Lab_7/Lab_7/bin/Debug/net6.0/Lab_7  
Ошибка: Массив должен быть двумерным и иметь размерность 2x2.  
Process finished with exit code 0.
```

Рисунок 1 – Вывод в консоль с некорректными данными

ВЫВОД

В ходе выполнения лабораторной работы были изучены основные принципы перегрузки методов. Были изучены механизмы реализации полиморфизма в C#. Были перегружены математические операторы. Отточена и закреплена работа с методами и классами в C#.

