Министерство цифрового развития, связи и массовых коммуникаций

Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования «Сибирский государственный университет

телекоммуникаций и информатики» (СибГУТИ)

Кафедра прикладной математики и кибернетики

**Лабораторная работа №4**

**«Разработка и модульное тестирование класса Матрица на С#»**

Выполнил:

Студент группы ИП-017

Костин А.В.

Работу проверил:

ассистент кафедры ПМиК

Агалаков А.А.

Новосибирск 2023 г.

**Содержание**

[**1.** **Задание** 3](#_Toc153300437)

[**2.** **Исходный код программы** 6](#_Toc153300438)

[**2.1.** **Код программы** 6](#_Toc153300439)

[**2.2.** **Код тестов** 11](#_Toc153300440)

[**3.** **Результаты модульных тестов** 15](#_Toc153300441)

[**4.** **Вывод** 15](#_Toc153300442)

1. **Задание**

Разработайте класс Матрица (Matrix) для операций матричной алгебры в соответствии с предложенной ниже спецификацией требований.

Разработайте тестовые наборы для тестирования методов класса на основе по критерию С2 (путей).

Выполните модульное тестирование класса средствами модульного тестирования Visual Studio.

Выполните анализ покрытия кода методов тестами.

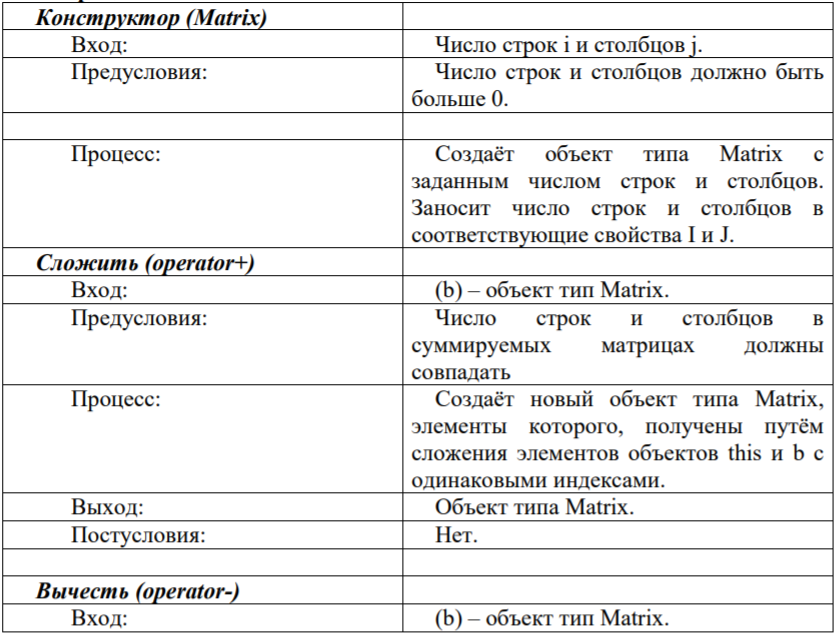
**Спецификация типа данных Матрица**

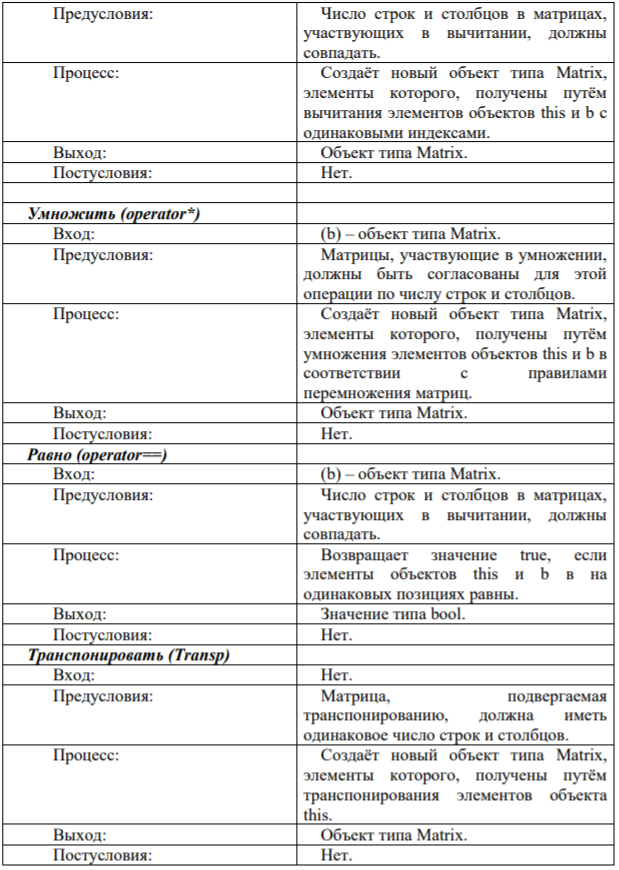
**Данные**

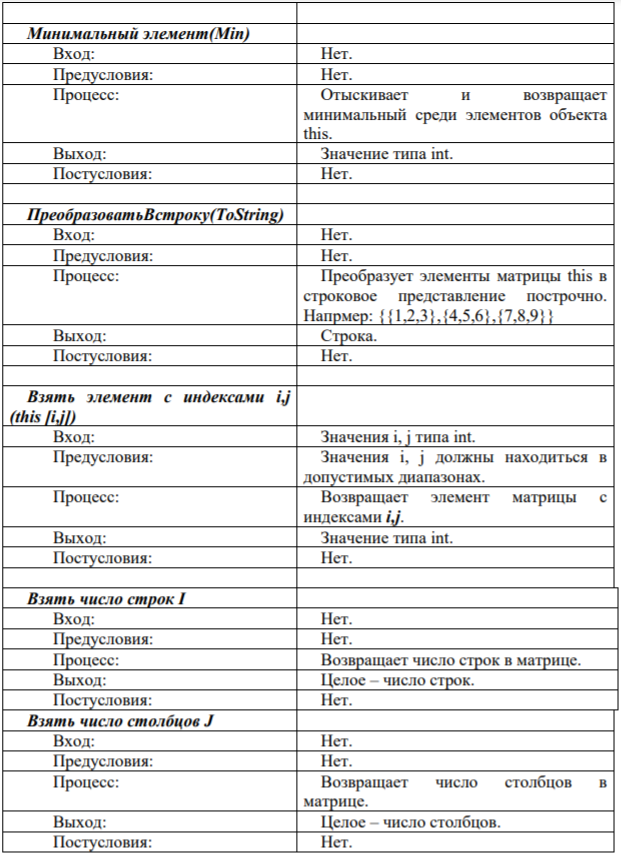
Матрица (тип Matrix) — это двумерная матрица со значениями целого типа.

Объект типа Матрица – не изменяемый.

**Операции**







1. **Исходный код программы**
   1. **Код программы**

***Program.cs***

using ModernCoding;

using System;

using System.Runtime.CompilerServices;

namespace ModernCoding

{

public class MyException : Exception

{

public MyException(string s) : base(s)

{ }

}

public class Matrix

{

int[,] m;

//Свойство для работы с числом строк.

public int row { get; }

//Свойство для работы с числом столбцов.

public int col { get; }

//Конструктор.

public Matrix(int i, int j)

{

if (i <= 0)

throw new MyException($"недопустимое значение строки = {i}");

if (j <= 0)

throw new MyException($"недопустимое значение столбца = {j}");

row = i;

col = j;

m = new int[i, j];

}

//Индексатор для доступа к значениям компонентов матрицы.

public int this[int i, int j]

{

get

{

if (i < 0 | i > row - 1)

throw new MyException($"неверное значение i = {i}");

if (j < 0 | j > col - 1)

throw new MyException($"неверное значение j = {j}");

return m[i, j];

}

set

{

if (i < 0 | i > row - 1)

throw new MyException($"неверное значение i = {i}");

if (j < 0 | j > col - 1)

throw new MyException($"неверное значение j = {0}");

m[i, j] = value;

}

}

//Сложение матриц.

public static Matrix operator +(Matrix a, Matrix b)

{

if (a.row != b.row || a.col!=b.col)

throw new MyException($"не одинаковый размер матриц");

Matrix res = new Matrix(a.row, a.col);

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

for (int j = 0; j < a.col; j++)

res[i, j] = a.m[i, j] + b.m[i, j];

return res;

}

public static Matrix operator -(Matrix a, Matrix b)

{

if (a.row != b.row || a.col != b.col)

throw new MyException($"не одинаковый размер матриц");

Matrix res = new Matrix(a.row, a.col);

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

for (int j = 0; j < a.col; j++)

res[i, j] = a.m[i, j] - b.m[i, j];

return res;

}

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

{

if (a.row != b.col || a.col != b.row)

throw new MyException($"не согласованный размер матриц");

Matrix res = new Matrix(a.row, a.col);

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

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

for (int k = 0; k < b.row; k++)

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

return res;

}

public Matrix transp()

{

Matrix res = new Matrix(this.col, this.row);

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

{

for (int j = 0; j < this.row; j++)

{

res[i, j] = this.m[j, i];

}

}

return res;

}

public int min()

{

int m = this.m[0,0];

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

for (int j = 0; j < this.col; j++)

if (this.m[i, j] < m)

m = this.m[i, j];

return m;

}

public string toString()

{

string res = "";

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

{

for (int j = 0; j < this.col; j++)

{

res += this.m[i, j].ToString();

if (j != this.col - 1)

res += " ";

}

if (i != this.row - 1)

res += ", ";

}

return res;

}

public static bool operator ==(Matrix a, Matrix b)

{

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

for (int j = 0; j < a.col; j++)

if (a[i, j] != b[i, j])

return false;

return true;

}

public static bool operator !=(Matrix a, Matrix b)

{

return !(a == b);

}

//Вывод значений компонентов на консоль.

public void Show()

{

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

{

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

{

Console.Write("\t" + this[i, j]);

}

Console.WriteLine();

}

Console.WriteLine();

}

public override bool Equals(object obj)

{

return (this as Matrix) == (obj as Matrix);

}

}

}

* 1. **Код тестов**

***UnitTest1.cs***

using System;

using Microsoft.VisualStudio.TestTools.UnitTesting;

using ModernCoding;

namespace MatrixTests

{

[TestClass]

public class MatrixTests

{

[TestMethod]

[ExpectedException(typeof(MyException))]

public void Matrix\_Expected\_MyException\_i()

{

//act (выполнить)

Matrix a = new Matrix(0, 2);

}

[TestMethod]

[ExpectedException(typeof(MyException))]

public void Matrix\_Expected\_MyException\_j()

{

//act (выполнить)

Matrix a = new Matrix(2, -1);

}

[TestMethod]

[ExpectedException(typeof(MyException))]

public void this\_Expected\_MyException\_set\_j()

{

//act (выполнить)

Matrix a = new Matrix(2, 2);

a[1, 3] = 2;

}

[TestMethod]

[ExpectedException(typeof(MyException))]

public void this\_Expected\_MyException\_get\_i()

{

//act (выполнить)

Matrix a = new Matrix(2, 2);

int r = a[3, 1];

}

[TestMethod]

public void Equel()

{

//arrange(обеспечить)

Matrix a = new Matrix(2, 2);

a[0, 0] = 1; a[0, 1] = 1; a[1, 0] = 1; a[1, 1] = 1;

Matrix b = new Matrix(2, 2);

b[0, 0] = 1; b[0, 1] = 1; b[1, 0] = 1; b[1, 1] = 1;

//act (выполнить)

//bool r = a == b;

//assert(доказать)

//Assert.IsTrue(r);

Assert.AreEqual(a, b);

}

[TestMethod]

public void Summa()

{

//arrange(обеспечить)

Matrix a = new Matrix(2, 2);

a[0, 0] = 1; a[0, 1] = 1; a[1, 0] = 1; a[1, 1] = 1;

Matrix b = new Matrix(2, 2);

b[0, 0] = 2; b[0, 1] = 2; b[1, 0] = 2; b[1, 1] = 2;

Matrix expected = new Matrix(2, 2);

expected[0, 0] = 3; expected[0, 1] = 3;

expected[1, 0] = 3; expected[1, 1] = 3;

Matrix actual = new Matrix(2, 2);

//act (выполнить)

actual = a + b;

//assert(доказать)

Assert.IsTrue(actual == expected);//Оракул

}

[TestMethod]

public void Dif()

{

//arrange(обеспечить)

Matrix a = new Matrix(2, 2);

a[0, 0] = 1; a[0, 1] = 1;

a[1, 0] = 1; a[1, 1] = 1;

Matrix b = new Matrix(2, 2);

b[0, 0] = 2; b[0, 1] = 2;

b[1, 0] = 2; b[1, 1] = 2;

Matrix expected = new Matrix(2, 2);

expected[0, 0] = 1; expected[0, 1] = 1;

expected[1, 0] = 1; expected[1, 1] = 1;

Matrix actual = new Matrix(2, 2);

//act (выполнить)

actual = b - a;

//assert(доказать)

Assert.IsTrue(actual == expected);//Оракул

}

[TestMethod]

public void Multiply()

{

//arrange(обеспечить)

Matrix a = new Matrix(2, 2);

a[0, 0] = 1; a[0, 1] = 1; a[1, 0] = 1; a[1, 1] = 1;

Matrix b = new Matrix(2, 2);

b[0, 0] = 2; b[0, 1] = 2; b[1, 0] = 2; b[1, 1] = 2;

Matrix expected = new Matrix(2, 2);

expected[0, 0] = 4; expected[0, 1] = 4;

expected[1, 0] = 4; expected[1, 1] = 4;

Matrix actual = new Matrix(2, 2);

//act (выполнить)

actual = a \* b;

//assert(доказать)

Assert.IsTrue(actual == expected);//Оракул

}

[TestMethod]

public void transpose()

{

//arrange(обеспечить)

Matrix a = new Matrix(2, 2);

a[0, 0] = 1; a[0, 1] = 2; a[1, 0] = 3; a[1, 1] = 4;

Matrix expected = new Matrix(2, 2);

expected[0, 0] = 1; expected[0, 1] = 3;

expected[1, 0] = 2; expected[1, 1] = 4;

Matrix actual = new Matrix(2, 2);

//act (выполнить)

actual = a.transp();

//assert(доказать)

Assert.IsTrue(actual == expected);//Оракул

}

[TestMethod]

public void min()

{

//arrange(обеспечить)

Matrix a = new Matrix(2, 2);

a[0, 0] = 1; a[0, 1] = 2; a[1, 0] = 3; a[1, 1] = 4;

int expected = 1;

//act (выполнить)

int actual = a.min();

//assert(доказать)

Assert.IsTrue(actual == expected);//Оракул

}

[TestMethod]

public void mtos()

{

//arrange(обеспечить)

Matrix a = new Matrix(2, 2);

a[0, 0] = 1; a[0, 1] = 2; a[1, 0] = 3; a[1, 1] = 4;

string expected = "1 2, 3 4";

//act (выполнить)

string actual = a.toString();

//assert(доказать)

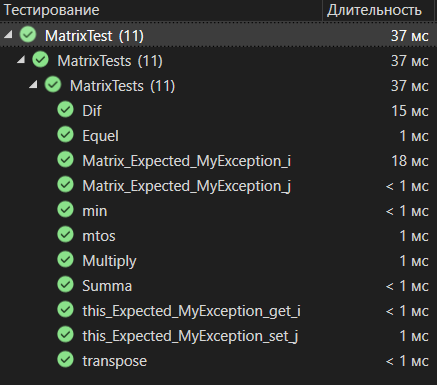
Assert.IsTrue(actual == expected);//Оракул

}

}

}

1. **Результаты модульных тестов**

****

1. **Вывод**

По итогам данной лабораторной работе были сформированы практические навыки разработки на С# и модульного тестирования классов средствами Visual Studio.