Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики» (СибГУТИ)

Кафедра ПМиК

**Лабораторная работа № 4**

**по дисциплине «Современные технологии программирования»**

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

Выполнил: студент 4 курса ф. ИВТ, гр. ИП-813

Пещеров Вячеслав Александрович

Проверил: ассистент

к. ПМиК Агалаков А.А.

Новосибирск, 2021

Оглавление

[**Задание** 3](#_Toc90043845)

[**Реализация** 4](#_Toc90043846)

[**Демонстрация работы** 5](#_Toc90043847)

[**Вывод** 6](#_Toc90043848)

[**Список литературы** 7](#_Toc90043849)

[**Приложение** 8](#_Toc90043850)

[Листинг 1 Matrix.cs 8](#_Toc90043851)

[Листинг 2. UnitTest1.cs 13](#_Toc90043852)

# **Задание**

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

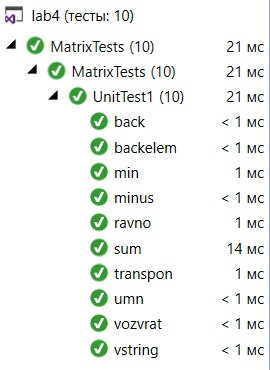
Разработайте тестовые наборы для тестирования методов класса на основе по критерию С2 (путей).

Выполните модульное тестирование класса средствами модульного тестирования Visual Studio. Выполните анализ покрытия кода методов тестами

# **Реализация**

В данной работе мы создали класс матрицы и все необходимые для работы с ним операции, такие как: конструктор, сложить, вычесть, умножить, равно, транспонировать, минимальный элемент, преобразовать в строку, взять элемент с индексами, взять число строк I, взять число столбцов J.

# **Демонстрация работы**



# **Вывод**

Мы научились работать в среде Visual Studio, а именно разрабатывать в ней модульные тесты для тестирования наших функции и классов на языке C#. Данная среда отлично подходит для выполнения модульного тестирования и автоматизации.

# **Список литературы**

1. Подбельский В.В., Фомин С.С.инт Курс программирования на языке Си: учебник. – М.:ДМК Пресс, 2012 – 384 с.
2. Павловская Т.А. C#. Программирование на языке высокого уровня: Учебник для вузов. - СПб. : Питер, 2014 - 432 с. : ил. - (Серия "Учебник для вузов").
3. Рихтер Дж. CLR via C#. Программирование на платформе Microsoft.NET Framework 4 на языке C# . 3-е изд.: - СПб.:Питер, 2012 - 928 с. : ил.
4. Котляров, В. П. Основы тестирования программного обеспечения : учебное пособие для СПО / В. П. Котляров. — Саратов : Профобразование, 2019 — 335 c. — ISBN 978-5-4488-0364-2. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/86202.html (дата обращения: 21.08.2020). — Режим доступа: для авторизир. пользователей

# **Приложение**

## Листинг 1 Matrix.cs

using System;

namespace lab4

{

public class MyException : Exception //обработка исключений

{

public MyException(string s) : base(s)

{ }

}

public class Matrix

{

int[,] m;

public int I { get; }

public int J { get; }

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 }");

I = i;

J = j;

m = new int[i, j];

}

public int this[int i, int j]

{

get

{

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

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

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

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

return m[i, j];

}

set

{

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

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

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

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

m[i, j] = value;

}

}

public static Matrix operator +(Matrix a, Matrix b)// сложение матриц

{

Matrix c = new Matrix(a.I, a.J);

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

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

{

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

}

return c;

}

public static Matrix operator -(Matrix a, Matrix b)//вычитание матриц

{

Matrix c = new Matrix(a.I, a.J);

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

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

{

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

}

return c;

}

public static Matrix operator \*(Matrix a, Matrix b)// перемножение матриц

{

Matrix c = new Matrix(a.I, a.J);

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

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

{

c[i,j] = 0;

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

c[i, j]+= a.m[i, k] \* b.m[k, j];

}

return c;

}

public static Matrix Transp(Matrix a)// транспонирование матрицы

{

int t;

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

{

for (int j = i; j < a.J; ++j)

{

t = a[i,j];

a[i,j] = a[j,i];

a[j,i] = t;

}

}

return a;

}

public static int Minimum(Matrix a)//нахождения минимума матрицы

{

int min = a[0,0];

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

{

for (int j = 1; j < a.J; ++j)

{

if (a[i, j-1] > a[i, j])

{

min = a[i, j];

}

}

}

return min;

}

public static string String(Matrix a)// матрица в строку

{

string str="";

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

{

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

{

str += a[i, j].ToString();

}

}

return str;

}

public static int index(Matrix a, int i, int j)//получить элемент

{

return a[i,j];

}

public static int Raw(Matrix a)//получить индекс строки

{

return a.I;

}

public static int Column(Matrix a)//получить индекс столбца

{

return a.J;

}

public static bool operator ==(Matrix a, Matrix b)//сравнивание матриц

{

bool q = true;

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

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

{

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

{

q = false; break;

}

}

return q;

}

public static bool operator !=(Matrix a, Matrix b)//сравнивание матриц на неравенство

{

return !(a == b);

}

public void Show()//вывод матрицы

{

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

{

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

{

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

}

Console.WriteLine();

}

Console.WriteLine();

}

public override bool Equals(object obj)

{

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

}

}

}

## Листинг 2. UnitTest1.cs

using System;

using Microsoft.VisualStudio.TestTools.UnitTesting;

using lab4;

namespace MatrixTests

{

[TestClass]

public class UnitTest1

{

[TestMethod]

public void sum()

{

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);

actual = a + b;

Assert.IsTrue(actual == expected);

}

[TestMethod]

public void minus()

{

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);

actual = a - b;

Assert.IsTrue(actual == expected);

}

[TestMethod]

public void umn()

{

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);

actual = a \* b;

Assert.IsTrue(actual == expected);

}

[TestMethod]

public void ravno()

{

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;

bool actual=false;

if (a == b)

actual=true;

Assert.AreEqual(actual,true);

}

[TestMethod]

public void transpon()

{

Matrix a = new Matrix(3, 3);

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

a[1, 0] = 3; a[1, 1] = 4; a[1, 2] = 5;

a[2, 0] = 6; a[2, 1] = 7; a[2, 2] = 8;

Matrix b = new Matrix(3, 3);

b[0, 0] = 0; b[0, 1] = 3; b[0, 2] = 6;

b[1, 0] = 1; b[1, 1] = 4; b[1, 2] = 7;

b[2, 0] = 2; b[2, 1] = 5; b[2, 2] = 8;

Matrix actual = new Matrix(3, 3);

actual = Matrix.Transp(a);

Assert.AreEqual(b, actual);

}

[TestMethod]

public void min()

{

Matrix a = new Matrix(3, 3);

a[0, 0] = 9; a[0, 1] = 1; a[0, 2] = 2;

a[1, 0] = 3; a[1, 1] = 4; a[1, 2] = 5;

a[2, 0] = 6; a[2, 1] = 7; a[2, 2] = 8;

int expected = 1;

int actual = 0;

actual= Matrix.Minimum(a);

Assert.AreEqual(expected, actual);

}

[TestMethod]

public void vstring()

{

Matrix a = new Matrix(3, 3);

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

a[1, 0] = 3; a[1, 1] = 4; a[1, 2] = 5;

a[2, 0] = 6; a[2, 1] = 7; a[2, 2] = 8;

string expected = "012345678";

string actual = "";

actual = Matrix.String(a);

Assert.AreEqual(expected, actual);

}

[TestMethod]

public void backelem()

{

Matrix a = new Matrix(3, 3);

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

a[1, 0] = 3; a[1, 1] = 4; a[1, 2] = 5;

a[2, 0] = 6; a[2, 1] = 7; a[2, 2] = 8;

int expected = 5;

int actual = 0;

actual = Matrix.index(a,1,2);

Assert.AreEqual(expected, actual);

// Assert.IsTrue(actual==b);

}

[TestMethod]

public void back()

{

Matrix a = new Matrix(2, 3);

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

a[1, 0] = 3; a[1, 1] = 4; a[1, 2] = 5;

int expected = 2;

int actual = 0;

actual = Matrix.Raw(a);

Assert.AreEqual(expected, actual);

// Assert.IsTrue(actual==b);

}

[TestMethod]

public void vozvrat()

{

Matrix a = new Matrix(2, 3);

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

a[1, 0] = 3; a[1, 1] = 4; a[1, 2] = 5;

int expected = 3;

int actual = 0;

actual = Matrix.Column(a);

Assert.AreEqual(expected, actual);

// Assert.IsTrue(actual==b);

}

}

}