Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Сибирский государственный университет телекоммуникаций и информатики»
(СибГУТИ)

Лабораторная работа №2

«Модульное тестирование библиотеки классов на языке C# средствами VisualStudio»

Вариант №4

Выполнил: студент 4 курса
ИВТ, гр. ИП-013
Копытина Т.А.
Проверил: ассистент кафедры
ПМиК
Агалаков А.А.

Новосибирск, 2023 г.

Цель

Сформировать практические навыки разработки модульных тестов для библиотек классов C# и выполнения модульного тестирования с помощью средств автоматизации VisualStudio.

Задание

Разработайте на языке C# класс, содержащий функции в соответствии с вариантом задания. Разработайте тестовые наборы данных для тестирования функций класса, по критерию C1. Протестируйте созданный класс с помощью средств автоматизации модульного тестирования VisualStudio. Проанализируйте результаты выполненных тестов по объёму покрытия тестируемого кода. Напишите отчёт о результатах проделанной работы.

1. Поиск максимума из трёх чисел
2. Функция получает двумерный массив вещественных переменных A. Отыскивает и возвращает произведение значений компонентов массива, у которых сумма значений индексов – чётная.
3. Функция получает двумерный массив вещественных переменных A. Отыскивает и возвращает минимальное значение компонентов массива, лежащих на и ниже главной диагонали

Реализация

В ходе выполнения задания был реализован класс с функциями в соответствии с заданием. Далее подробнее о каждом из реализованных методе:

public static int MaxNumber(inta, intb, intc) – функцияполучаетнавходтричисла, вычисляется максимальное из них и возвращается.

public static double MultOfOddInd(double[,] arr) – функция получает на вход двумерный вещественный массив, идет цикл по всему массиву и если сумма индексов четная – считается произведение. Результат произведения возвращается.

public static double MinElemDiag(double[,] arr) – функция получает на вход двумерный вещественный массив, идет цикл по всему массиву и если элемент принадлежит главной диагонали или находится ниже ее, то он сравнивается с минимальным элементом на данный момент. Возвращает минимальный элемент.

Рис. 1 – демонстрация работоспособности реализованных функций.

Так же были реализованы тесты всех методов по критерию С1 – набор тестов в совокупности должен обеспечить прохождение каждой ветви не менее одного раза.

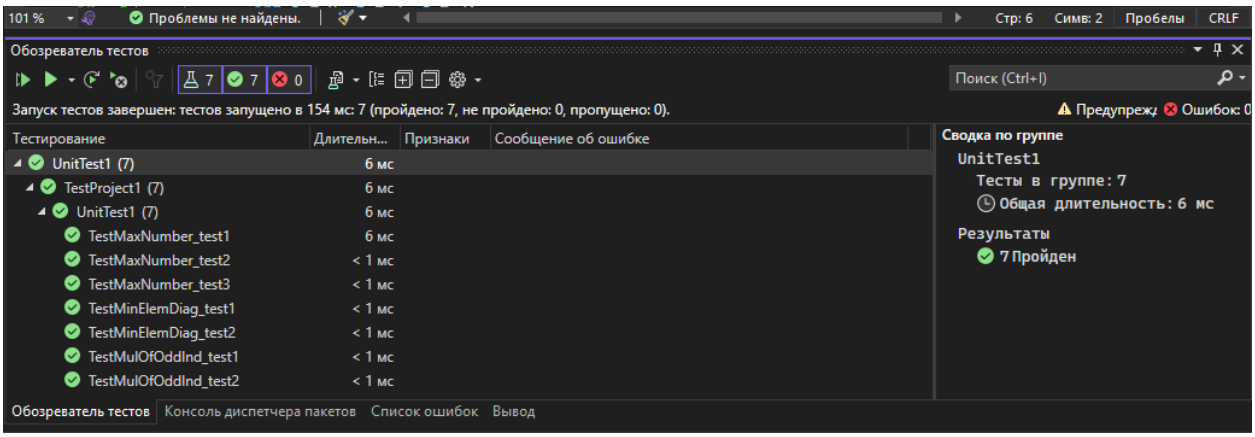


Рис. 2 – демонстрация результатов проведенного тестирования по критерию С1.

В конце НЕ были получены результаты выполненных тестов по объёму покрытия тестируемого кода, так данная проверка доступна лишь в Enterprise версии VisualStudio.

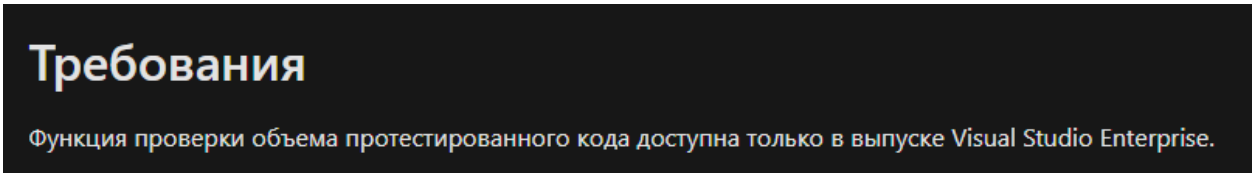


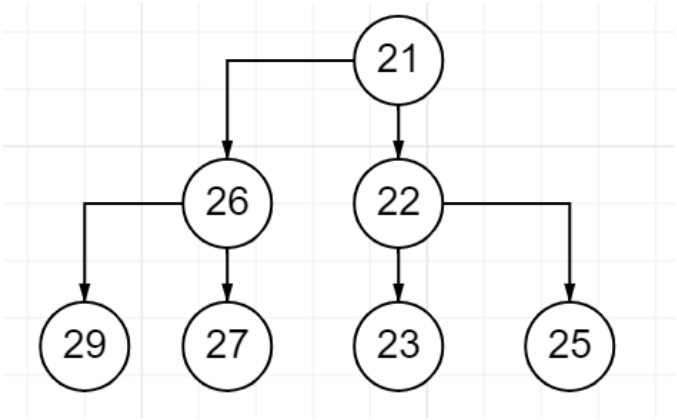
Рис.3 – доказательство, информация с официальной документации VisualStudioc сайта Microsoft.

УГП и тестовые наборы данных

Для функции public static intMaxNumber(int a, int b, int c):

```
21. if (a > b)
22. if (a > c)
23. return a;
24. else
25. return c;
26. elseif (b > c)
27. return b;
28. else
29. return c;
```

УГП:



Содержит следующие ветви:21-22,21-26,22-23,22-25, 26-27, 26-29 и пути:21-22-23, 21-22-25,21-26-27,21-26-29.

Тестовые наборы, которые подойдут, по критерию С1:

1.a=7,b=2,c =3;(покроем ветви 21-22,22-23);

2.a=1,b=2,c =-3;(покроем ветви 21-22,22-25);

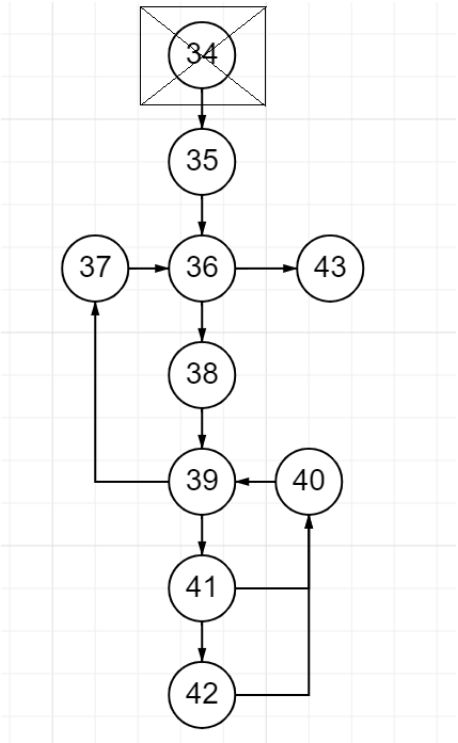
3.a=-7,b=2,c =-3;(покроем ветви 21-26,26-27);

3.a=-7,b=2,c =3;(покроем ветви 21-26,26-29);

<u>Для функции</u> public static double MultOfOddInd(double[,] mas):

```
34. doubleresult = 1;
35. for (int i = 0;
36. i <mas.GetLength(0);
37. i++)
38. for (int j = 0;
39. j <mas.GetLength(1);
40. j++)
41. if ((i + j) % 2 == 0)
42. result *= mas[i, j];
43. returnresult;
```

УГП:



Содержит следующие ветви: 34-35-36,36-38-39,36-43,39-41,41-40,41-42-40,39-37-36 и
пути 34-35-36-43,34-35-36-38-39-37-36-43,34-35-36-38-39-41-40-37-36-43,34-35-36-38-39-41-42-40-37-36-43.

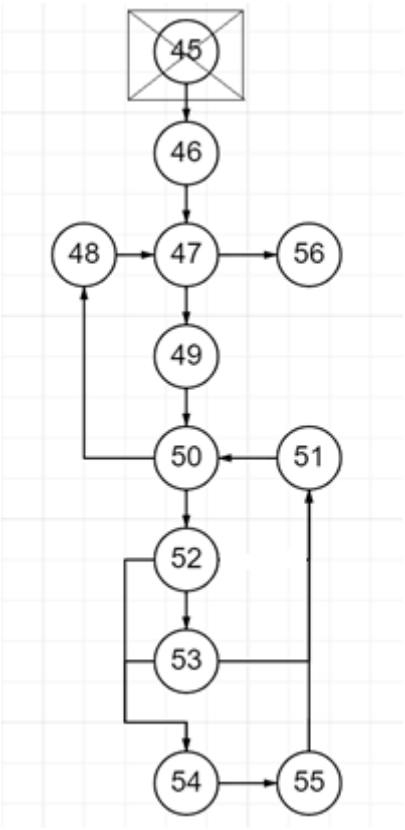
Тестовый набор, который подойдет, по критерию С1:

double[,] mas={{1.2,3.4}, {5.6,7.8}};

Для функции public static intMinElemDiag(int a, int b, int c):

```
45. doubleresult = float.MaxValue;
46. for (int i = 0;
47. i <mas.GetLength(0);
48. i++)
49. for (int j = 0;
50. j <mas.GetLength(1);
51. j++)
52. if (i == j
53. || i > j)
54. if (mas[i, j] < result)
55. result = mas[i, j];
56. returnresult;
```

УГП:



Содержит следующие ветви: 45-46-47,47-49-50,47-56, 50-48-47,50-52,52-51,52-53,52-54-55-51, 53-51, 53-54-55-51 и

пути 45-46-47-56, 45-46-47-49-50-48-56, 45-46-47-49-50-52-53-51-50-48-56, 45-46-47-49-50-52-54-55-51-50-48-56,45-46-47-49-50-52-53-54-55-51-50-48-56.

Тестовый набор, который подойдет, по критерию C1:

double[,] mas={{ 1.2,3.4}, { 5.6,7.8}};

Вывод

Были сформированы практические навыки разработки модульных тестов для библиотек классов C# и выполнения модульного тестирования с помощью средств автоматизации VisualStudio, разработан класс на языке C#, содержащий функции в соответствии с вариантом задания, разработаны тестовые наборы данных для тестирования функций класса, по критерию C1, не были получены результаты покрытия кода тестами в связи с ограничением версии VisualStudio.

Листингпрограммы:

Program.cs:

```
using System;

namespace lab2
{
    public class Program
    {
        static void Main(string[] args)
        {
int a = 7, b = 2, c = 3;
double[,] mas1 = { { 1, 2, 3.5 }, { 4.2, 5, 6.7 } };
double[,] mas2 = { { 1.6, 2, -2.4 }, { 0.15, 1.5, 15 }, { 0.54, -0.3, 5.42 }
};
Console.WriteLine("Максимальный элемент: " + MaxNumber(a, b, c));
Console.WriteLine("Результат перемножения: " + MultOfOddInd(mas1));
Console.WriteLine("Минимальный элемент: " + MinElemDiag(mas2));


        }

        public static intMaxNumber(int a, int b, int c)
        {
            if (a > b)
                if (a > c)
                    return a;
                else
                    return c;
            else if (b > c)
                return b;
            else
                return c;
        }

        public static double MultOfOddInd(double[,] mas)
        {
            double result = 1;
            for (int i = 0; i <mas.GetLength(0); i++)
                for (int j = 0; j <mas.GetLength(1); j++)
                    if ((i + j) % 2 == 0)
                        result *= mas[i, j];
            return result;

        }

        public static double MinElemDiag(double[,] mas)
        {
            double result = float.MaxValue;
            for (int i = 0; i <mas.GetLength(0); i++)
                for (int j = 0; j <mas.GetLength(1); j++)
                    if (i == j || i > j)
                        if (mas[i, j] < result)
                            result = mas[i, j];
            return result;
        }
    }
}
```

UnitTest1.cs:

```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using lab2;
using System;

namespace TestProject1
{
    [TestClass]
    public class UnitTest1
    {
        [TestMethod]
        public void TestMaxNumber_test1()
        {
            int a = 7, b = 2, c = 3;
            int expected = 7;
            int result = Program.MaxNumber(a, b, c);
            Assert.AreEqual(expected, result);
        }

        [TestMethod]
        public void TestMaxNumber_test2()
        {
            int a = -7, b = 2, c = -3;
            int expected = 2;
            int result = Program.MaxNumber(a, b, c);
            Assert.AreEqual(expected, result);
        }

        [TestMethod]
        public void TestMaxNumber_test3()
        {
            int a = -7, b = 2, c = 3;
            int expected = 3;
            int result = Program.MaxNumber(a, b, c);
            Assert.AreEqual(expected, result);
        }

        [TestMethod]
        public void TestMulOfOddInd_test1()
        {
            double[,] mas = { { 1, 2, 3.5 }, { 4.2, 5, 6.7 } };
            double expected = 17.5;
            double result = Program.MultOfOddInd(mas);
            Assert.AreEqual(expected, result);
        }

        [TestMethod]
        public void TestMulOfOddInd_test2()
        {
            double[,] mas = { { 1, 300002, 1 }, { 300002, 1, 300002 } };
            double expected = 1;
            double result = Program.MultOfOddInd(mas);
            Assert.AreEqual(expected, result);
        }

        [TestMethod]
        public void TestMinElemDiag_test1()
        {
            double[,] mas = { { 1.6, 2, -2.4 }, { 0.15, 1.5, 15 }, { 0.54, -
0.3, 5.42 } };
            double expected = -0.3;
            double result = Program.MinElemDiag(mas);
            Assert.AreEqual(expected, result);
```

```
        }

        [TestMethod]
        public void TestMinElemDiag_test2()
        {
            double[,] mas = { { 1.6, 2, -2.4 }, { 0.15, 1.5, -150 }, { 0.54,
0.3, 5.42 } };
            double expected = 0.15;
            double result = Program.MinElemDiag(mas);
Assert.AreEqual(expected, result);
        }
    }
}
```