Федеральное агентство связи

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет

телекоммуникаций и информатики»

**Лабораторная работа №2**

**Вариант №4**

**«Модульное тестирование библиотеки классов на C# средствами Visual Studio»**

Выполнил: студент IV курса

ИВТ, гр. ИП-713

Михеев Н.А.

Проверил: ассистент кафедры

ПМиК

Агалаков А.А.

Новосибирск, 2020 г.

**Цель**

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

**Задание**

Разработайте на языке С# класс, содержащий функции в соответствии с вариантом задания. Разработайте тестовые наборы данных для тестирования функций класса, по критерию С1. Протестируйте созданный класс с помощью средств автоматизации модульного тестирования Visual Studio. 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 Max\_Number(int a, int b, int c) – функция получает на вход три числа, вычисляется максимальное из них и возвращается.

public static double MulOfOddInd(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.

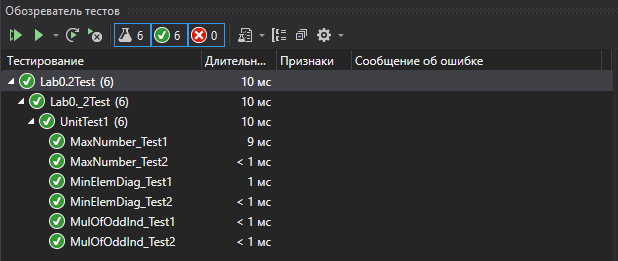


Рис. 2 – демонстрация результатов проведенного тестирования по критерию С1.

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

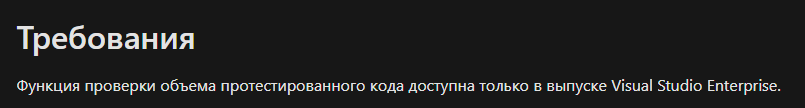
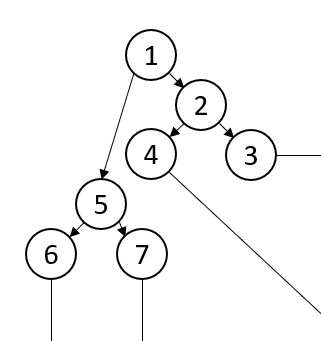
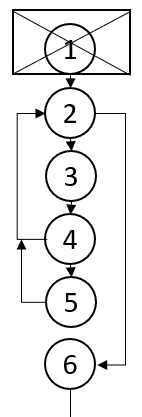


Рис.3 – доказательство, информация с официальной документации Visual Studio с сайта Microsoft.

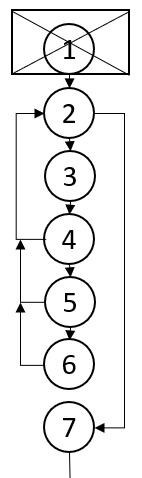
Так же были построены управляющие графы программы для всех функий. Для public static int Max\_Number(int a, int b, int c):



Для public static double MulOfOddInd(double[,] arr):



Для последней функции public static double MinElemDiag(double[,] arr):



**Вывод**

Были сформированы практические навыки разработки модульных тестов для библиотек классов C# и выполнения модульного тестирования с помощью средств автоматизации Visual Studio, разработан класс на языке C#, содержащий функции в соответствии с вариантом задания, разработаны тестовые наборы данных для тестирования функций класса, по критерию С1, не были получены результаты покрытия кода тестами в связи с ограничением версии Visual Studio.

**Листинг программы:**

**Program.cs:**

using System;

using System.Diagnostics.CodeAnalysis;

namespace Lab0.\_2

{

public class Program

{

public static int Max\_Number(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 MulOfOddInd(double[,] arr)

{

double res = 1;

for (int i = 0; i < arr.GetLength(0); i++)

for (int j = 0; j < arr.GetLength(1); j++)

if ((i + j) % 2 == 0)

res \*= arr[i, j];

return res;

}

public static double MinElemDiag(double[,] arr)

{

double res = float.MaxValue;

for (int i = 0; i < arr.GetLength(0); i++)

for (int j = 0; j < arr.GetLength(1); j++)

if (i == j || i > j)

if (arr[i, j] < res)

res = arr[i, j];

return res;

}

[ExcludeFromCodeCoverage]

static void Main()

{

Console.WriteLine("Max number: " + Max\_Number(7, 2, 3));

Console.WriteLine("Result of multiply: " + MulOfOddInd(new double[,] { { 2.3, 6.2, 1.3 }, { 6.25, 14.5, 2.2 } }));

Console.WriteLine("Minimal element: " + MinElemDiag(new double[,] { { 1.6, 2, -2 }, { 0.15, 1.2, 6 }, { 0.22, 1.13, 6.22 } }));

#if DEBUG

Console.WriteLine("Press any key to exit.");

Console.ReadKey();

#endif

}

}

}

**UnitTest1.cs:**

using System;

using Microsoft.VisualStudio.TestTools.UnitTesting;

using Lab0.\_2;

namespace Lab0.\_2Test

{

[TestClass]

public class UnitTest1

{

[TestMethod]

public void MaxNumber\_Test1()

{

int expect = 25;

int result = Program.Max\_Number(6, 25, -100);

Assert.AreEqual(expect, result);

}

[TestMethod]

public void MaxNumber\_Test2()

{

int expect = -6;

int result = Program.Max\_Number(-6, -25, -100);

Assert.AreEqual(expect, result);

}

[TestMethod]

public void MulOfOddInd\_Test1()

{

double expect = 43.355;

double result = Program.MulOfOddInd(new double[,] { { 2.3, 6.2, 1.3 }, { 6.25, 14.5, 2.2 } });

Assert.AreEqual(expect, result);

}

[TestMethod]

public void MulOfOddInd\_Test2()

{

double expect = 8;

double result = Program.MulOfOddInd(new double[,] { { 2, 346346.436342, 2 }, { 36346.515, 2, 2.223523 } });

Assert.AreEqual(expect, result);

}

[TestMethod]

public void MinElemDiag\_Test1()

{

double expect = 0.15;

double result = Program.MinElemDiag(new double[,] { { 1.6, 2, -2 }, { 0.15, 1.2, 6 }, { 0.22, 1.13, 6.22 } });

Assert.AreEqual(expect, result);

}

[TestMethod]

public void MinElemDiag\_Test2()

{

double expect = -1.6;

double result = Program.MinElemDiag(new double[,] { { -1.6, 2, -2 }, { 0.15, 1.2, 6 }, { 0.22, 1.13, 6.22 } });

Assert.AreEqual(expect, result);

}

}

}