Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет

телекоммуникаций и информатики»

Кафедра ПМиК

Современные технологии программирования 1

Лабораторная работа №1.

Модульное тестирование программ на языке С# средствами Visual Studio

Вариант №2

Выполнил: студент 4 курса

Ф. ИВТ, группа: ИП-711

Мартасов И. О.

Проверил: доцент кафедры ПМиК

Зайцев Михаил Георгиевич

Новосибирск, 2020

**Содержание**

1. Задание.
2. УГП и тестовые наборы данных для тестирования функций класса.
3. Исходные тексты программ на языке C#
4. Результаты выполнения модульных тестов.
5. Результаты покрытия разработанного кода тестами.
6. Выводы по выполненной работе.

**Задание**

Разработайте на языке С# класс, содержащий функции в соответствии с вариантом задания.

Разработайте тестовые наборы данных по критерию С0 для тестирования функций класса.

Протестируйте созданный класс с помощью средств автоматизации модульного тестирования Visual Studio.

Напишите отчёт о результатах проделанной работы. Варианты заданий 71

**Вариант № 2**

1. Функция получает два одномерных массив a, b одинаковой длины. Возвращает массив c полученный суммированием значений массивов a,b.
2. Функция получает одномерный массив вещественных переменных и целое – параметр сдвига. Функция изменяет массив циклическим сдвигом значений его элементов влево на число позиций, равное параметру сдвига.
3. Функция находит и возвращает индекс начала первого вхождения последовательности целых чисел, представленных массивом int[] seq в другую последовательность, представленную массивом int[] vec.

**УГП и тестовые наборы данных для тестирования функций класса**

**Функция 1 (Count\_mas):**

**(A, B, C) = {({1,2,3,4,5 вх. массив}, {1,2,3,4,5 вх. массив},n= 5, {2,4,6,8,10 вых. массив})}**

**Функция 2 (Cicle\_rotate):**

**(A, B) = {({1,2,3,4,5 вх. массив}, n= 3, {4,5,1,2,3 вых. массив})}**

**Функция 3 (Sequence):**

**(A, B) = {({1,2,3,4,5 вх.массив}, {2,3 вх. Массив}, индекс = 1)}**

**Исходные тексты программ на языке C#**

**MyClass.cs**

using System;

using System.Buffers;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApp1

{

public static class MyClass

{

public static int[] Count\_mas(int[] a, int[] b, int n)

{

int[] c = new int[n];

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

{

c[i] = a[i] + b[i];

}

return c;

}

public static int Cicle\_rotate(double[] actual, int offset)

{

int n = actual.Length;

offset %= n;

Array.Reverse(actual);

Array.Reverse(actual, 0, n - offset);

Array.Reverse(actual, n - offset, offset);

return 0;

}

public static int Sequence(int[] seq, int[] vec, int seq\_n, int vec\_n)

{

string vec\_1 = string.Join("", vec);

string seq\_1 = string.Join("", seq);

int result = vec\_1.IndexOf(seq\_1);

return result;

}

}

}

**UnitTest1.cs**

using System;

using Microsoft.VisualStudio.TestTools.UnitTesting;

namespace TestMyClass

{

[TestClass]

public class UnitTestMyClass

{

[TestMethod]

public void Test\_count\_mas\_with\_correct\_length()

{

int n = 10;

int[] a = new int[n];

int[] b = new int[n];

int[] actual;

int[] expected = new int[n];

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

{

a[i] = b[i] = (i + 1);

expected[i] = a[i] + b[i];

}

actual = ConsoleApp1.MyClass.Count\_mas(a, b, n);

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

{

Assert.AreEqual(expected[i], actual[i]);

}

}

[TestMethod]

public void Test\_cicle\_with\_correct\_offset()

{

int offset = 5;

const int n = 10;

int expected = 0;

double[] expected\_mas = new double[n]

{5.1, 6.1, 7.1, 8.1, 9.1, 0.1, 1.1, 2.1, 3.1, 4.1};

double[] actual = new double[n];

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

{

actual[i] = 0.1 + i;

}

Assert.AreEqual(expected, ConsoleApp1.MyClass.Cicle\_rotate(actual, offset));

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

{

Assert.AreEqual(expected\_mas[i], actual[i]);

}

}

[TestMethod]

public void Test\_cicle\_with\_offset\_0()

{

int offset = 0;

const int n = 10;

int expected = 0;

double[] expected\_mas = new double[n]

{0.1, 1.1, 2.1, 3.1, 4.1, 5.1, 6.1, 7.1, 8.1, 9.1};

double[] actual = new double[n];

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

{

actual[i] = 0.1 + i;

}

Assert.AreEqual(expected, ConsoleApp1.MyClass.Cicle\_rotate(actual, offset));

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

{

Assert.AreEqual(expected\_mas[i], actual[i]);

}

}

[TestMethod]

public void Test\_cicle\_with\_offset\_bigger\_than\_n()

{

int offset = 14;

const int n = 10;

int expected = 0;

double[] expected\_mas = new double[n]

{4.1, 5.1, 6.1, 7.1, 8.1, 9.1, 0.1, 1.1, 2.1, 3.1};

double[] actual = new double[n];

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

{

actual[i] = 0.1 + i;

}

Assert.AreEqual(expected, ConsoleApp1.MyClass.Cicle\_rotate(actual, offset));

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

{

Assert.AreEqual(expected\_mas[i], actual[i]);

}

}

[TestMethod]

public void Test\_sequence\_entrance\_in\_end\_of\_vector()

{

const int seq\_n = 3;

const int vec\_n = 10;

int actual;

int expected = 7;

int[] vector = new int[vec\_n]

{7, 8, 7, 3, 6, 7, 8, 7, 8, 9};

int[] sequence = new int[seq\_n]

{7, 8, 9};

actual = ConsoleApp1.MyClass.Sequence(sequence, vector, seq\_n, vec\_n);

Assert.AreEqual(expected, actual);

}

[TestMethod]

public void Test\_sequence\_with\_entrance\_in\_middle()

{

const int seq\_n = 4;

const int vec\_n = 10;

int actual;

int expected = 3;

int[] vector = new int[vec\_n]

{7, 8, 7, 3, 6, 7, 8, 7, 8, 7};

int[] sequence = new int[seq\_n]

{3, 6, 7, 8};

actual = ConsoleApp1.MyClass.Sequence(sequence, vector, seq\_n, vec\_n);

Assert.AreEqual(expected, actual);

}

[TestMethod]

public void Test\_sequence\_with\_entrance\_in\_beginning()

{

const int seq\_n = 2;

const int vec\_n = 10;

int actual;

int expected = 0;

int[] vector = new int[vec\_n]

{1, 8, 7, 3, 6, 7, 8, 7, 8, 7};

int[] sequence = new int[seq\_n]

{1, 8};

actual = ConsoleApp1.MyClass.Sequence(sequence, vector, seq\_n, vec\_n);

Assert.AreEqual(expected, actual);

}

[TestMethod]

public void Test\_sequence\_with\_no\_entrance()

{

const int seq\_n = 4;

const int vec\_n = 10;

int actual;

int expected = -1;

int[] vector = new int[vec\_n]

{1, 8, 7, 3, 7, 8, 7, 7, 8, 7};

int[] sequence = new int[seq\_n]

{7, 8, 7, 8};

actual = ConsoleApp1.MyClass.Sequence(sequence, vector, seq\_n, vec\_n);

Assert.AreEqual(expected, actual);

}

[TestMethod]

public void Test\_sequence\_with\_full\_entrance()

{

const int seq\_n = 10;

const int vec\_n = 10;

int actual;

int expected = 0;

int[] vector = new int[vec\_n]

{1, 8, 7, 3, 6, 7, 8, 7, 9, 7};

int[] sequence = new int[seq\_n]

{1, 8, 7, 3, 6, 7, 8, 7, 9, 7};

actual = ConsoleApp1.MyClass.Sequence(sequence, vector, seq\_n, vec\_n);

Assert.AreEqual(expected, actual);

}

[TestMethod]

public void Test\_sequence\_with\_one\_element\_entrance()

{

const int seq\_n = 1;

const int vec\_n = 10;

int actual;

int expected = 2;

int[] vector = new int[vec\_n]

{1, 8, 7, 3, 6, 7, 8, 7, 9, 7};

int[] sequence = new int[seq\_n]

{7};

actual = ConsoleApp1.MyClass.Sequence(sequence, vector, seq\_n, vec\_n);

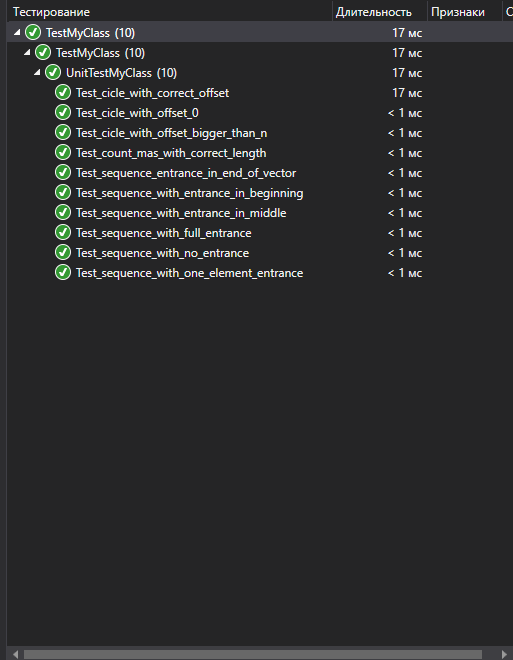
Assert.AreEqual(expected, actual);

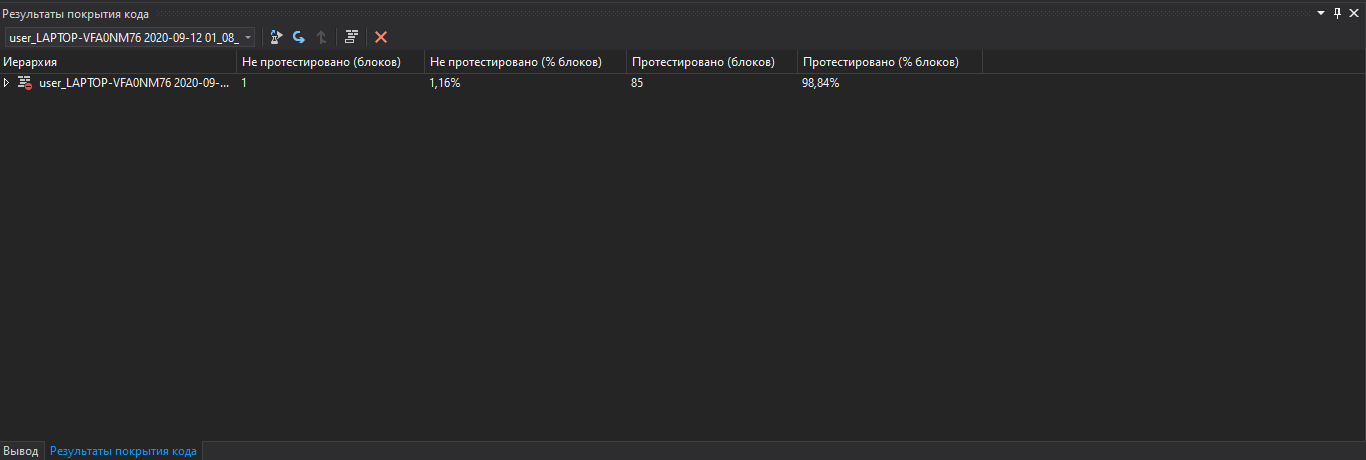
}

}

}

**Результаты выполнения модульных тестов и результаты покрытия разработанного кода тестами**

****

****

**Выводы по выполненной работе**

Были сформированы практические навыки разработки модульных тестов для тестирования функций классов и выполнения модульного тестирования на языке С# с помощью средств автоматизации Visual Studio