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 №1**

**«Модульное тестирование программ на языке С# средствами Visual Studio»**

**Вариант №4**

Выполнил: студент 4 курса

ИВТ, гр. ИП-113

Шпилев Д. И.

Проверил: старший преподаватель кафедры ПМиК

Агалаков А.А.

Новосибирск, 2024 г.

**Цель**

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

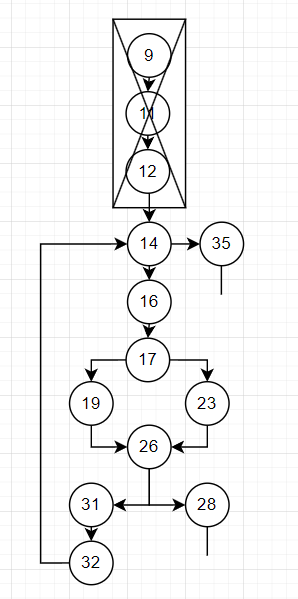
**Задание**

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.

1. Функция получает целое число b – основание системы счисления и строку s, содержащую представление целой части числа в системы счисления с основанием b. Функция формирует и возвращает из строки s целое число.
2. Функция получает одномерный массива целых переменных. Вычисляет и возвращает максимальный по значению элемент этого массива и номер его индекса.
3. Функция получает одномерный массив целых переменных. Вычисляет и возвращает максимальное значение среди нечётных элементов массива с нечётными значениями индекса и значение индекса (через параметр)

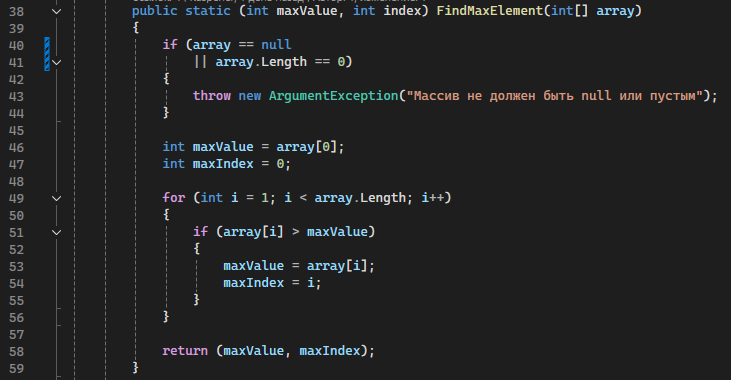
**УГП и тестовые наборы данных для тестирования функций класса**

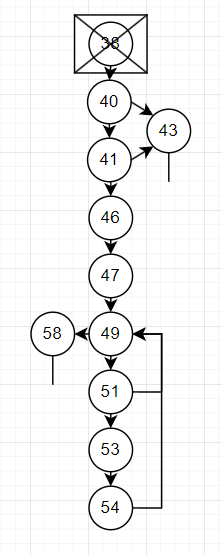


****

Для полного тестирования нужно пройти пути 17-19-26 и 17-23-26, а также попасть в исключение 26-28

Для прохождения путей 17-19-26 и 17-23-26 можно взять 16-ричное число, допустим «153AF». Для попадания в исключение 26-28 берем число «1254236» и систему счисления 2.

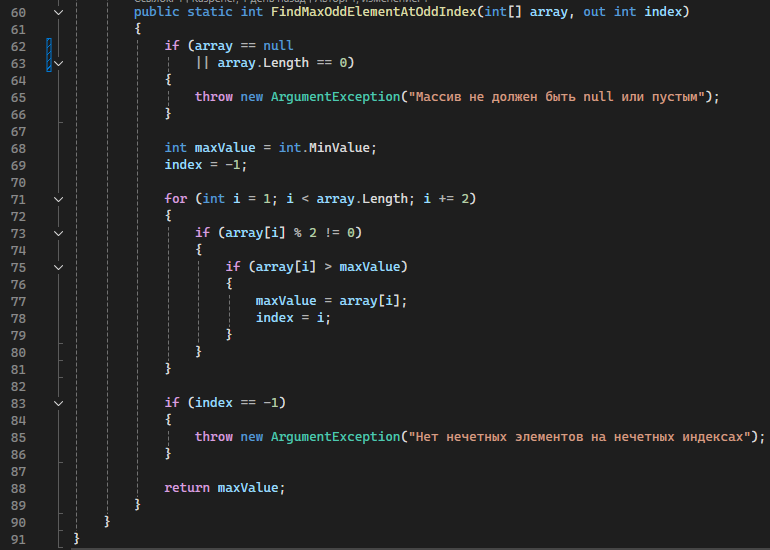
****

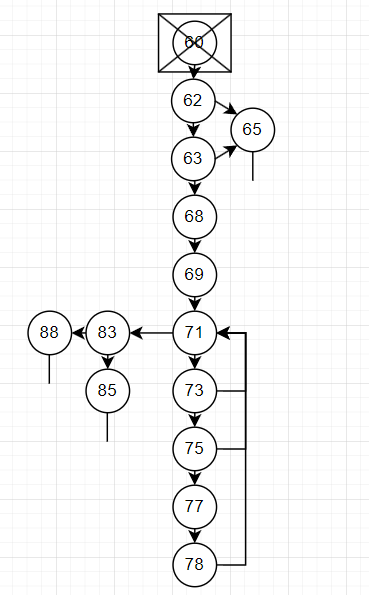
****

Для полного тестирования нужно выполнить исключения из пунктов 40 и 41 и попасть в пункт 43, а также попасть в путь 49-51-53-54.

Для исключения берем array = null и пустой массив array.

Для пути 49-51-53-54 можем взять array = { 3, 6, 23, 2, 56, 7, 4, 3, 2 };





Для полного тестирования нужно выполнить исключения из пунктов 62 и 63

и попасть в пункт 65, пройти путь 71-73-75-77-78, попасть в исключение 85 и выйти из программы 85.

Для исключения берем array = null и пустой массив array

Для выхода с исключением: array = { 3, 6, 23, 2, 56, 8, 4, 12, 2 };

Для выхода без исключения: array = { 3, 6, 23, 2, 56, 7, 4, 3, 2 };

**Листинг программы:**

**Program.cs:**

using System;

namespace Lab1

{

public class Program

{

static void Main(string[] args) { }

public static int ConvertToDecimal(int b, string s)

{

int result = 0;

int power = 1;

for (int i = s.Length - 1; i >= 0; i--)

{

int digitValue;

if (char.IsDigit(s[i]))

{

digitValue = s[i] - '0';

}

else

{

digitValue = char.ToUpper(s[i]) - 'A' + 10;

}

if (digitValue >= b)

{

throw new ArgumentException($"Цифра '{s[i]}' недопустима для системы счисления с основанием {b}");

}

result += digitValue \* power;

power \*= b;

}

return result;

}

public static (int maxValue, int index) FindMaxElement(int[] array)

{

if (array == null || array.Length == 0)

{

throw new ArgumentException("Массив не должен быть null или пустым");

}

int maxValue = array[0];

int maxIndex = 0;

for (int i = 1; i < array.Length; i++)

{

if (array[i] > maxValue)

{

maxValue = array[i];

maxIndex = i;

}

}

return (maxValue, maxIndex);

}

public static int FindMaxOddElementAtOddIndex(int[] array, out int index)

{

if (array == null || array.Length == 0)

{

throw new ArgumentException("Массив не должен быть null или пустым");

}

int maxValue = int.MinValue;

index = -1;

for (int i = 1; i < array.Length; i += 2)

{

if (array[i] % 2 != 0)

{

if (array[i] > maxValue)

{

maxValue = array[i];

index = i;

}

}

}

if (index == -1)

{

throw new ArgumentException("Нет нечетных элементов на нечетных индексах");

}

return maxValue;

}

}

}

**UnitTest1.cs:**

using Microsoft.VisualStudio.TestTools.UnitTesting;

using System;

namespace UnitTestLab1

{

[TestClass]

public class TestingConbert

{

[TestMethod]

public void ConvertHEX()

{

//arrange

int footing = 16;

string number = "153AF";

//act

int result = Lab1.Program.ConvertToDecimal(footing, number);

//assert

Assert.AreEqual(86959, result);

}

[TestMethod]

public void ConvertBIN()

{

//arrange

int footing = 2;

string number = "10001110";

//act

int result = Lab1.Program.ConvertToDecimal(footing, number);

//assert

Assert.AreEqual(142, result);

}

[TestMethod]

public void ConvertDEC()

{

//arrange

int footing = 10;

string number = "1254236";

//act

int result = Lab1.Program.ConvertToDecimal(footing, number);

//assert

Assert.AreEqual(1254236, result);

}

[TestMethod]

public void ConvertException()

{

//arrange

int footing = 2;

string number = "1254236";

//act

//assert

Assert.ThrowsException<ArgumentException>(() => Lab1.Program.ConvertToDecimal(footing, number));

}

}

[TestClass]

public class TestFindMaxElement

{

[TestMethod]

public void FindMax()

{

//arrange

int[] array = { 3, 6, 23, 2, 56, 7, 4, 3, 2 };

//act

var result = Lab1.Program.FindMaxElement(array);

//assert

Assert.AreEqual((56, 4), result);

}

[TestMethod]

public void FindMaxFromEquals()

{

//arrange

int[] array = { 3,3,3,3,3,3,3,3,3,3 };

//act

var result = Lab1.Program.FindMaxElement(array);

//assert

Assert.AreEqual((3, 0), result);

}

[TestMethod]

public void FindMaxExceptionZeroLenght()

{

//arrange

int[] array = { };

//act

//assert

Assert.ThrowsException<ArgumentException>(() => Lab1.Program.FindMaxElement(array));

}

[TestMethod]

public void FindMaxExceptionNullArray()

{

//arrange

//act

//assert

Assert.ThrowsException<ArgumentException>(() => Lab1.Program.FindMaxElement(null));

}

}

[TestClass]

public class TestFindMaxOddElement

{

[TestMethod]

public void FindMax()

{

//arrange

int[] array = { 3, 6, 23, 2, 56, 7, 4, 3, 2 };

int index;

//act

int value = Lab1.Program.FindMaxOddElementAtOddIndex(array, out index);

//assert

Assert.AreEqual((7, 5), (value, index));

}

[TestMethod]

public void NoFindMax()

{

//arrange

int[] array = { 3, 6, 23, 2, 56, 8, 4, 12, 2 };

int index;

//act

//assert

Assert.ThrowsException<ArgumentException>(() => Lab1.Program.FindMaxOddElementAtOddIndex(array, out index), "Нет нечетных элементов на нечетных индексах");

}

[TestMethod]

public void FindMaxExceptionZeroLenght()

{

//arrange

int[] array = { };

int index;

//act

//assert

Assert.ThrowsException<ArgumentException>(() => Lab1.Program.FindMaxOddElementAtOddIndex(array, out index), "Массив не должен быть null или пустым");

}

[TestMethod]

public void FindMaxExceptionNullArray()

{

//arrange

int index;

//act

//assert

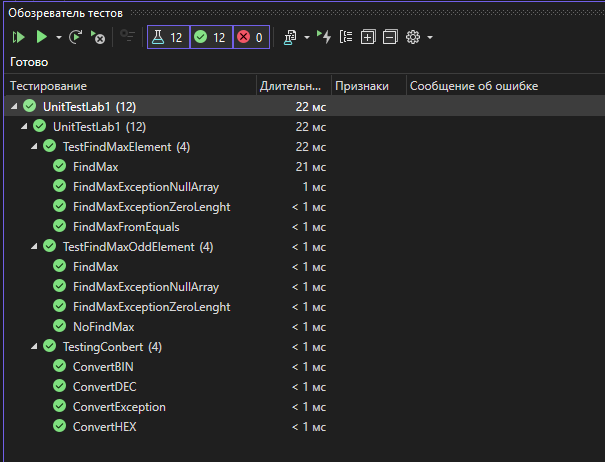
Assert.ThrowsException<ArgumentException>(() => Lab1.Program.FindMaxOddElementAtOddIndex(null, out index), "Массив не должен быть null или пустым");

}

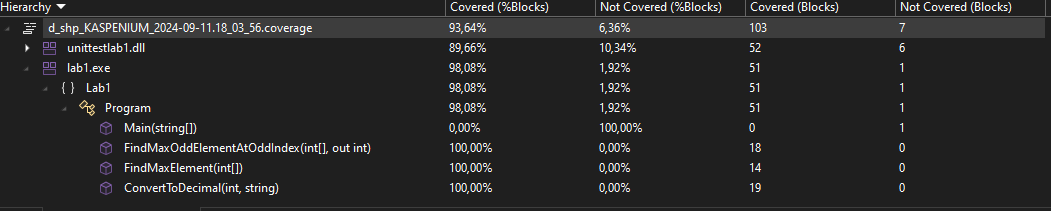
}

}

**Результаты выполнения модульных тестов**

****

**Результаты покрытия разработанного кода тестами.**



**Вывод**

В результате работы над лабораторной работой были сформированы практические навыки разработки функций классов на языке С#, разработка модульных тестов для тестирования функций классов и выполнения модульного тестирования на языке С# с помощью средств автоматизации Visual Studio.