МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ

ХАРЬКОВСКИЙ НАЦИОНАЛЬНЫЙ УНИВЕРСИТЕТ

РАДИОЭЛЕКТРОНИКИ

Кафедра ИИ

Отчет

о выполнении лабораторной работы № 1

по дисциплине «Пна C#»

Название работы: Управляемая тестами разработка классов.

Цель работы: Изучение особенностей разработки классов в среде Visual Studio 2019 на языке C# с использованием техники TDD

(Test Driven Development).

Задание № 41

Выполнил ст. гр. ІТШІ-19-4 :

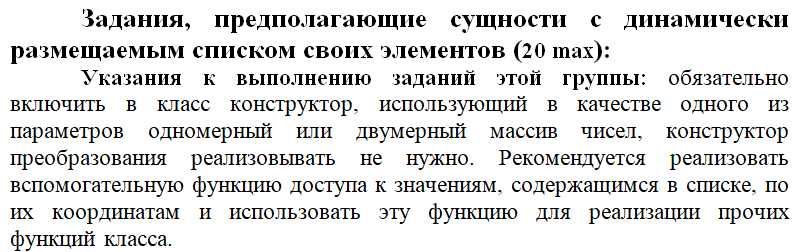
Зубков Артем Олегович

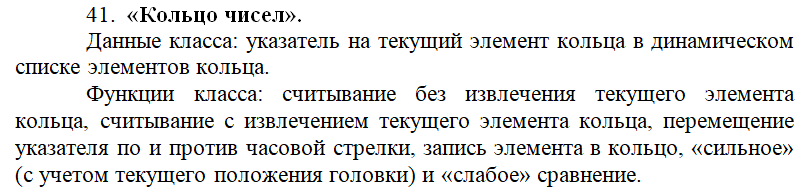
Принял:

Бибичков Игорь Евгеньевич

2020

Задание:





1.Класс с пустыми функциями (кроме функций с консольным вводом/выводом)

public class Ring

{

public class Element

{

public double number;

public Element next;

public Element previous;

public int key;

}

public Element Cur { get; set; }

public Ring()

{

Cur = null;

}

public Ring(double[] arr)

{

foreach(double num in arr)

{

Add(num);

}

}

public Ring(Ring myRing)

{

Cur = null;

Element myElement = myRing.Cur;

while(myRing.Cur!=null)

{

Add(myRing.Cur.number);

myRing.Right();

if (myRing.Cur == myElement)

break;

}

}

public double Input()

{

Console.Write("Введите число: ");

double num = Convert.ToDouble(Console.ReadLine());

return num;

}

public void Output()

{

Console.WriteLine("Данное число с индексом {0} равно: {1}", Cur.key, Cur.number);

}

public double Access()

{

Element cur1 = Cur;

int maxi=-1;

while (cur1.key>maxi)

{

maxi = cur1.key;

cur1 = cur1.next;

}

int mykey;

do

{

Console.Write("Введите действительный индекс числа в диапазоне от 0 до {0}: ",maxi);

mykey = Convert.ToInt32(Console.ReadLine());

} while (mykey<0 || mykey>maxi);

cur1 = Cur;

do

{

if (mykey == cur1.key)

break;

cur1 = cur1.next;

} while (cur1 != Cur || cur1 !=null);

return cur1.number;

}

public void Add(double num)

{

throw new Exception();

}

public void Right()

{

throw new Exception();

}

public void Left()

{

throw new Exception();

}

public bool EaseEquals(Element obj)

{

throw new Exception();

}

public bool DiffEquals(Element obj, Element obj1)

{

throw new Exception();

}

public Element Read()

{

throw new Exception();

}

public Element ReadAndRetrive()

{

throw new Exception();

}

}

2. Реализация тестов

[TestClass]

public class UnitTest1

{

[TestMethod]

//сравнение чисел текущего(Cur) элемента кольца и другого элемента того же класса Element при равных значениях;

public void EaseEquals\_With\_Same\_Number\_Are\_Numbers\_Equal()

{

//Arrange

double a = 1;

double b = 1;

Ring myring = new Ring();

myring.Cur = new Ring.Element();

Ring.Element second = new Ring.Element();

bool expected = true;

//Act

myring.Cur.number = a;

second.number = b;

bool actual = myring.EaseEquals(second);

//Assert

Assert.AreEqual(expected, actual,"Numbers are equal");

}

[TestMethod]

//сравнение чисел текущего(Cur) элемента кольца и другого элемента того же класса Element при неравных значениях;

public void EaseEquals\_With\_Different\_Number\_Are\_Not\_Numbers\_Equal()

{

//Arrange

double a = 2;

double b = 7;

Ring myring = new Ring();

myring.Cur = new Ring.Element();

Ring.Element second = new Ring.Element();

bool expected = false;

//Act

myring.Cur.number = a;

second.number = b;

bool actual = myring.EaseEquals(second);

//Assert

Assert.AreEqual(expected, actual, "Numbers are not equal");

}

[TestMethod]

//проверка на правильность записи элемента в кольцо

public void Add\_To\_Ring\_Are\_added()

{

//Arrange

double a = 3;

Ring myring = new Ring();

//Act

myring.Add(a);

//Assert

Assert.AreEqual(a, myring.Cur.number, "Adding works right");

}

[TestMethod]

//проверка на правильность перемещения указателя по часовой стрелке

public void Right\_Current\_becomes\_next\_Is\_next()

{

//Arrange

Ring.Element first = new Ring.Element();

Ring.Element second = new Ring.Element();

Ring.Element third = new Ring.Element();

Ring myring = new Ring();

myring.Cur = third;

myring.Cur.next = first;

first.next = second;

second.next = myring.Cur;

//Act

myring.Right();

//Assert

Assert.AreEqual(first, myring.Cur, "Current pointer has moved to the right");

}

[TestMethod]

//проверка на правильность перемещения указателя против часовой стрелки

public void Left\_Current\_becomes\_previous\_Is\_previous()

{

//Arrange

Ring.Element first = new Ring.Element();

Ring.Element second = new Ring.Element();

Ring.Element third = new Ring.Element();

Ring myring = new Ring();

myring.Cur = third;

myring.Cur.previous = second;

second.previous = first;

first.previous = third;

//Act

myring.Left();

//Assert

Assert.AreEqual(second, myring.Cur, "Current pointer has moved to the left");

}

[TestMethod]

//сравнение чисел при том, что головка(Cur) указывает на один из элементов

public void DiffEquals\_Cur\_points\_to\_elem\_Not\_equal()

{

//Arrange

Ring.Element first = new Ring.Element();

first.number = 1;

Ring.Element second = new Ring.Element();

second.number = 1;

Ring myring = new Ring();

myring.Cur = first;

bool expected = false;

//Act

bool actual = myring.DiffEquals(first, second);

//Assert

Assert.AreEqual(expected, actual, "They are not equal");

}

[TestMethod]

//сравнение одинаковых чисел при том, что головка(Cur) не указывает ни на один из элементов

public void DiffEquals\_Numbers\_are\_sameANDcur\_does\_not\_point\_to\_elem\_Numbers\_are\_Equal()

{

//Arrange

Ring.Element first = new Ring.Element();

Ring.Element second = new Ring.Element();

double a = 1;

double b = 1;

Ring myring = new Ring();

bool expected = true;

//Act

first.number = a;

second.number = b;

bool actual = myring.DiffEquals(first, second);

//Assert

Assert.AreEqual(expected, actual, "They are equal");

}

[TestMethod]

//сравнение разных чисел при том, что головка(Cur) не указывает ни на один из элементов

public void DiffEquals\_Numbers\_are\_differentANDcur\_does\_not\_point\_to\_elem\_Numbers\_are\_Equal()

{

//Arrange

Ring.Element first = new Ring.Element();

Ring.Element second = new Ring.Element();

double a = 1;

double b = 2.2;

Ring myring = new Ring();

bool expected = false;

//Act

first.number = a;

second.number = b;

bool actual = myring.DiffEquals(first, second);

//Assert

Assert.AreEqual(expected, actual, "They are not equal");

}

[TestMethod]

//проверка на правильность считывания элемента, без его извлечения

public void Read\_CurrElement\_Is\_got\_right()

{

//Arrange

Ring myring = new Ring();

myring.Cur = new Ring.Element();

//Act

Ring.Element x = myring.Read();

//Assert

Assert.AreEqual(myring.Cur, x, "It is read right");

}

[TestMethod]

//проверка на правильность считывания элемента, при том, что он извлекается

public void ReadAndRetrive\_retrived\_currElement\_Is\_got\_right()

{

//Arrange

Ring myring = new Ring();

Ring.Element myelem = new Ring.Element();

//Act

myring.Cur = myelem;

Ring.Element x = myring.ReadAndRetrive();

//Assert

Assert.AreEqual(myelem, x, "It is read right");

}

[TestMethod]

//проверка на то, что элемент извлекся

public void ReadAndRetrive\_retrived\_currElement\_Is\_retrived()

{

//Arrange

Ring myring = new Ring();

Ring.Element myelem = new Ring.Element();

//Act

myring.Cur = myelem;

Ring.Element x = myring.ReadAndRetrive();

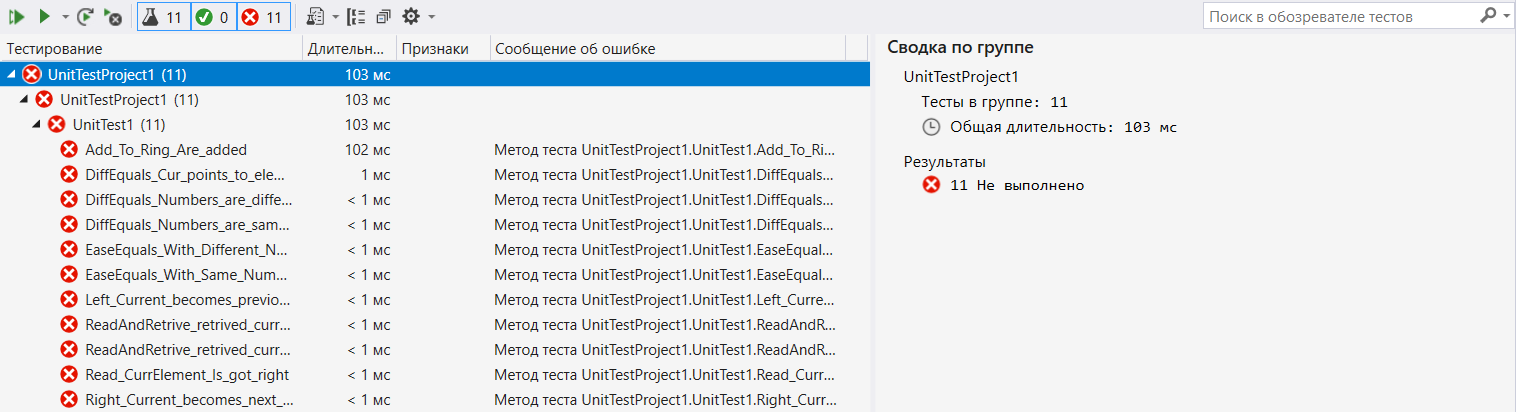
//Assert

Assert.AreNotEqual(myring.Cur, myelem, "Current element is retrived");

}

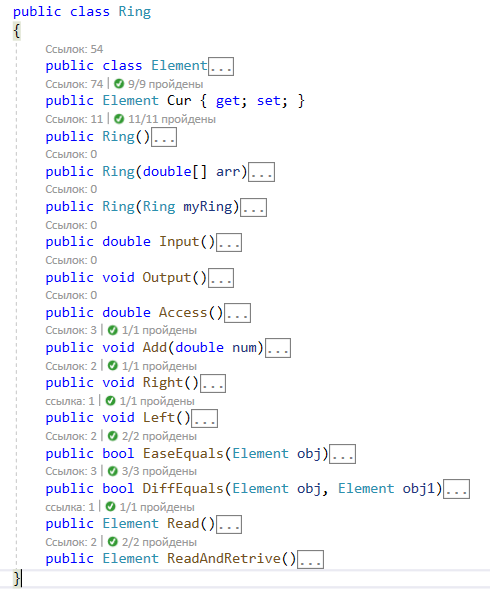
}

3. Скрины красного прохождения тестов



4. Финальная реализация класса

(прототипы и данные):



(весь код):

public class Ring

{

public class Element

{

public double number;

public Element next;

public Element previous;

public int key;

}

public Element Cur { get; set; }

public Ring()

{

Cur = null;

}

public Ring(double[] arr)

{

foreach(double num in arr)

{

Add(num);

}

}

public Ring(Ring myRing)

{

Cur = null;

Element myElement = myRing.Cur;

while(myRing.Cur!=null)

{

Add(myRing.Cur.number);

myRing.Right();

if (myRing.Cur == myElement)

break;

}

}

public double Input()

{

Console.Write("Введите число: ");

double num = Convert.ToDouble(Console.ReadLine());

return num;

}

public void Output()

{

Console.WriteLine("Данное число с индексом {0} равно: {1}", Cur.key, Cur.number);

}

public double Access()

{

Element cur1 = Cur;

int maxi=-1;

while (cur1.key>maxi)

{

maxi = cur1.key;

cur1 = cur1.next;

}

int mykey;

do

{

Console.Write("Введите действительный индекс числа в диапазоне от 0 до {0}: ",maxi);

mykey = Convert.ToInt32(Console.ReadLine());

} while (mykey<0 || mykey>maxi);

cur1 = Cur;

do

{

if (mykey == cur1.key)

break;

cur1 = cur1.next;

} while (cur1 != Cur || cur1 !=null);

return cur1.number;

}

public void Add(double num)

{

if(Cur == null)

{

Element first = new Element();

Cur = first;

Cur.number = num;

Cur.next = null;

Cur.previous = null;

Cur.key = 0;

}

else

{

Element myElement = new Element();

if(Cur.next == null)

{

Cur.next = myElement;

Cur.previous = myElement;

myElement.next = Cur;

myElement.previous = Cur;

}

else

{

myElement.next = Cur.next;

myElement.previous = Cur;

Cur.next.previous = myElement;

Cur.next = myElement;

}

Cur = myElement;

Cur.number = num;

Cur.key = Cur.previous.key + 1;

Element cur1 = Cur;

while(cur1.key <= cur1.next.key)

{

cur1.next.key += 1;

cur1 = cur1.next;

}

}

}

public void Right()

{

if (Cur.next != null)

Cur = Cur.next;

}

public void Left()

{

if (Cur.previous != null)

Cur = Cur.previous;

}

public bool EaseEquals(Element obj)

{

return Cur.number == obj.number;

}

public bool DiffEquals(Element obj, Element obj1)

{

if (Cur == obj || Cur == obj1 || obj.number != obj1.number)

return false;

else

return true;

}

public Element Read()

{

return Cur;

}

public Element ReadAndRetrive()

{

Element el = Cur;

if(Cur.next != Cur.previous)

{

Cur.previous.next = Cur.next;

Cur.next.previous = Cur.previous;

Cur = Cur.previous;

}

else

{

if(Cur.previous != null)

{

Cur = Cur.previous;

Cur.next = null;

Cur.previous = null;

}

else

{

Cur = null;

}

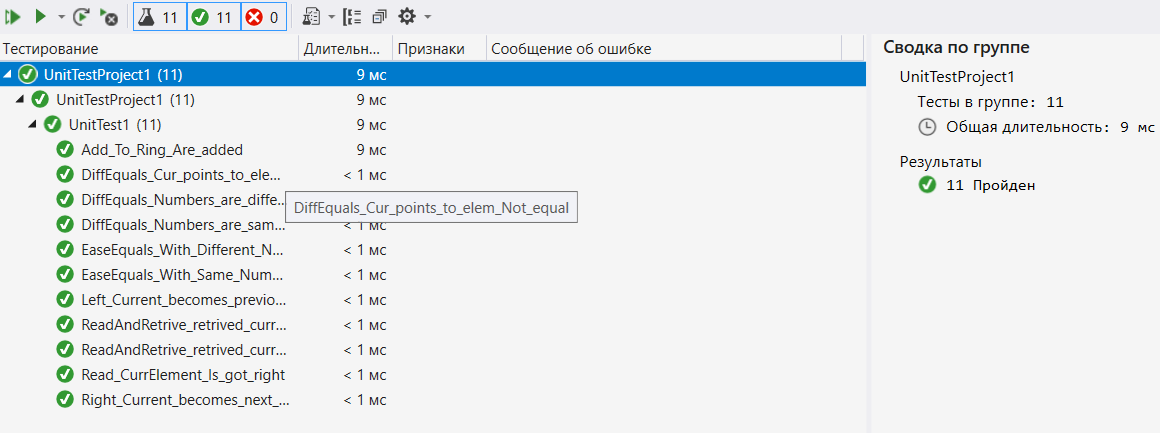
}

return el;

}

}

5.Скрины зеленого прохождения тестов



Выводы: Научился создавать проект тестов, реализовал все тест-функции, с использованием техники TDD, для каждой функции из задания по созданию класса “Кольцо чисел”. Протестировал функции с пустой реализацией(т.е с исключением) и с полной реализацией, получив при этом соответствующие результаты выполнения. Т.е. научился разрабатывать управляемые тестами классы.