Національний технічний університет України «КПІ»
Факультет інформатики та обчислювальної техніки
Кафедра Інформаційних систем та технологій

Лабораторна робота №2

з дисципліни « Сучасні технології розробки WEB-застосувань на платформі Microsoft.NET»

на тему: « Модульне тестування. Ознайомлення з засобами та практиками модульного тестування»

Виконала:
студентка гр. ІС-11
Гаврильчик Яна
Викладач:
Бардін В.

2023 рік

Мета лабораторної роботи – навчитися створювати модульні тести для вихідного коду розроблювального програмного забезпечення.

Завдання:

1. Додати до проекту власної узагальненої колекції (застосувати виконану лабораторну роботу No1) проект модульних тестів, використовуючи певний фреймворк (Nunit, Xunit, тощо).
2. Розробити модульні тести для функціоналу колекції.
3. Дослідити ступінь покриття модульними тестами вихідного коду колекції, використовуючи, наприклад, засіб AxoCover.

Варіант 4:

4	Дек (черга з двома кінцями)	Див. Queue<T>	Збереження даних за допомогою динамічно зв’язаного списку

Посилання на код GitHub:

https://github.com/melancholiya/dotnet_labs1-2

Хід виконання роботи:

Додано до проєкту власної узагальненої колекції проєкт модульних тестів, використовуючи фреймворк NUnit:

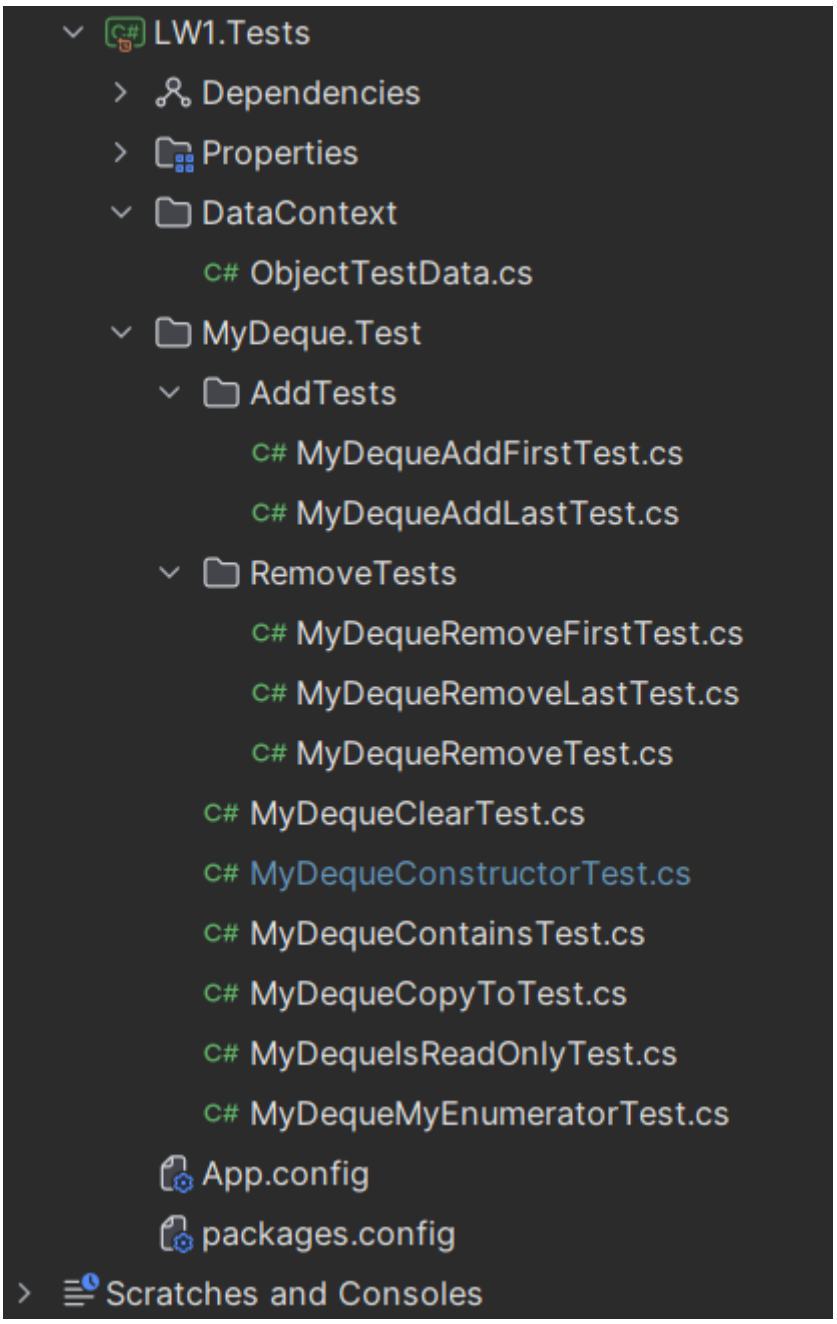

Рис.1 - Структура проєкту модульних тестів

Створено набір тестових даних для колекції `ObjectTestData:`

```
12 usages  45 tests OK  Yanok27
public static IEnumerable<TestCaseData> GetTestCasesWithValues()
{
    yield return new TestCaseData( deque: new DoubleEndedQueue<int>{1}, expectedItem: 1);
    yield return new TestCaseData( deque: new DoubleEndedQueue<int>{1,2,3,4}, expectedItem: 1);
}

8 usages  45 tests OK  Yanok27
public static IEnumerable<TestCaseData> GetTestCasesWithEmptyCollections()
{
    yield return new TestCaseData( deque: new DoubleEndedQueue<int>());
}

2 usages  45 tests OK  Yanok27
public static IEnumerable<TestCaseData> GetTestCasesWithOneElement()
{
    yield return new TestCaseData( deque: new DoubleEndedQueue<int>{1}, expectedItem: 1);
}

1 usage  45 tests OK  Yanok27
public static IEnumerable<TestCaseData> GetTestCasesForCopyToMethod()
{
    yield return new TestCaseData( deque: new DoubleEndedQueue<int>{1,2,3,4}, array: new int[6], expectedAr
}
6 usages  45 tests OK  Yanok27
public static IEnumerable<TestCaseData>GetTestCasesForEnumerator()
{
    yield return new TestCaseData( deque: new DoubleEndedQueue<int>{1,2,3,4});
}
                                                                                    AAA
```

Рис.2 - Набір тестових даних

Наведено деякі приклади із тестів до методів колекції:

```
//if count>0, then add item to the beginning
[Test, TestCaseSource(typeof(ObjectTestData) ,nameof(ObjectTestData.GetTestCasesWithValues))]
2 tests OK  Yanok27
public void GivenNonEmptyDeque_WhenAddFirst_ThenItemIsAddedToTheBeginning(TestCaseData testCaseData)
{
    //Arrange (Given)
    var deque :DoubleEndedQueue<int> = testCaseData.Deque;
    var expectedItem :int = testCaseData.ExpectedItem;
    //Act (When)
    deque.AddFirst(expectedItem);
    //Assert (Then)
    deque.Head.Value.Should().Be(expectedItem);
    deque.Tail.Should().NotBeNull();
}
```

Рис.3 - Тестування методу додавання елементу на початок черги

```
//remove should return false if item is not in deque
[Test, TestCaseSource(typeof(ObjectTestData), nameof(ObjectTestData.GetTestCasesWithEmptyCollections))]
1 test OK  Yanok27
public void Remove_NonExistingItem_ShouldReturnFalse(TestCaseData testCaseData)
{
    //Arrange
    var deque :DoubleEndedQueue<int> = testCaseData.Deque;
    //Act
    var actualResult :bool = deque.Remove( item: 2);
    //Assert
    actualResult.Should().BeFalse();
}
```

Рис.4 - Тестування методу видалення

```
[Test, TestCaseSource(typeof(ObjectTestData), nameof(ObjectTestData.GetTestCasesWithValues))]
2 tests OK  Yanok27
public void GivenNonEmptyDeque_WhenClear_ThenDequeIsEmpty(TestCaseData testCaseData)
{
    //Arrange (Given)
    var deque :DoubleEndedQueue<int> = testCaseData.Deque;
    //Act (When)
    deque.Clear();
    //Assert (Then)
    deque.Count.Should().Be(0);
    deque.Head.Should().BeNull();
    deque.Tail.Should().BeNull();
}
```

Рис.5 - Тестування очищення колекції

```
[Test, TestCaseSource(typeof(ObjectTestData), nameof(ObjectTestData.GetTestCasesWithValues))]
2 tests OK  Yanok27
public void GivenNonEmptyDeque_WhenContains_ThenItemIsNotContained(TestCaseData testCaseData)
{
    //Arrange (Given)
    var deque :DoubleEndedQueue<int> = testCaseData.Deque;
    //Act (When)
    var actualItem :bool = deque.Contains( item: 0);
    //Assert (Then)
    actualItem.Should().BeFalse();
}
```

Рис.6 - Тестування на вміст елементу у черзі

Ступінь покриття тестами вихідного коду (було виключено проєкт із самою реалізацією колекції та допоміжні класи для консольної роботи проєкту):

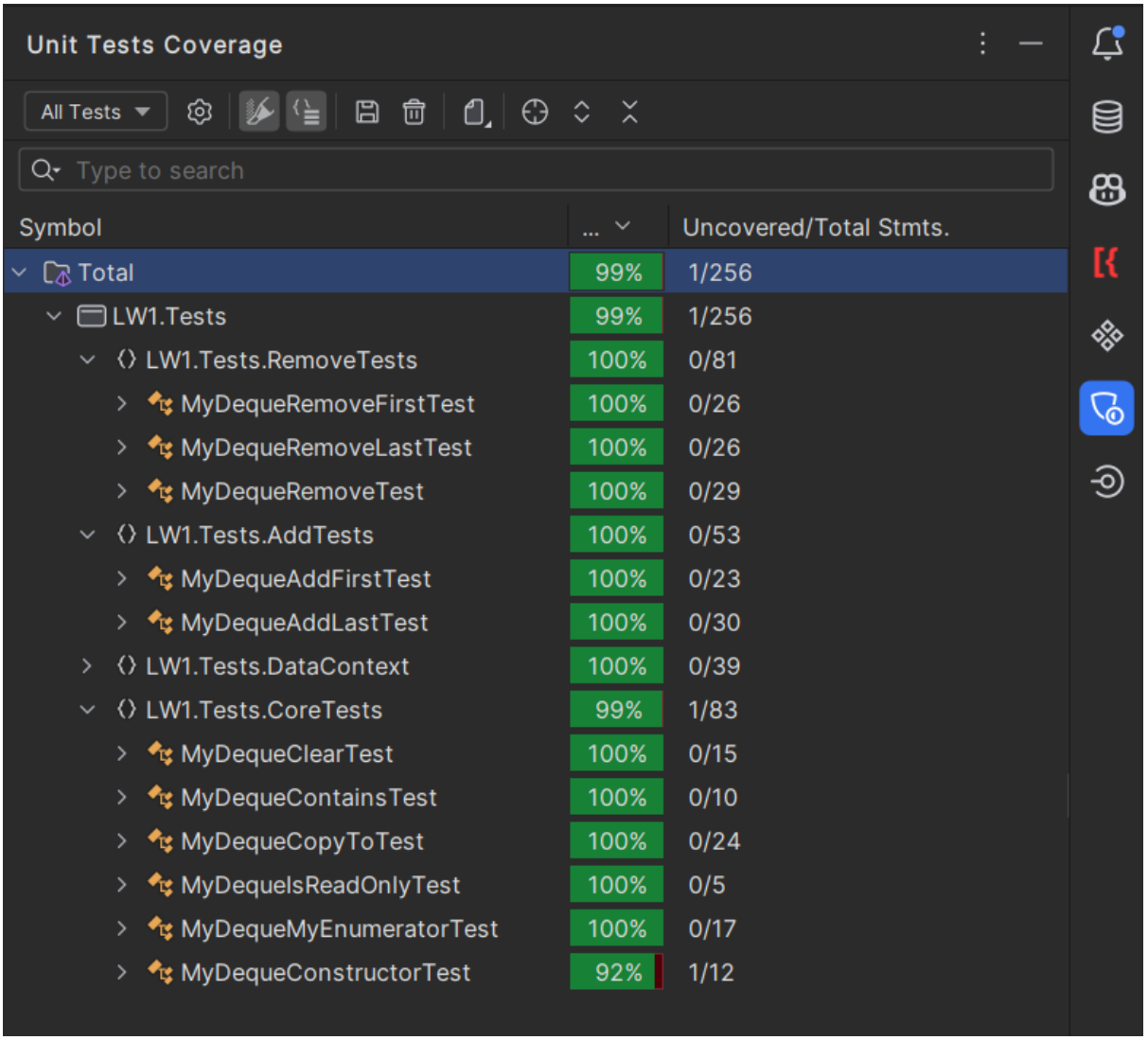

Рис.7 - Покриття тестами вихідного коду

Також згенеровано html-сторінку покриття коду:

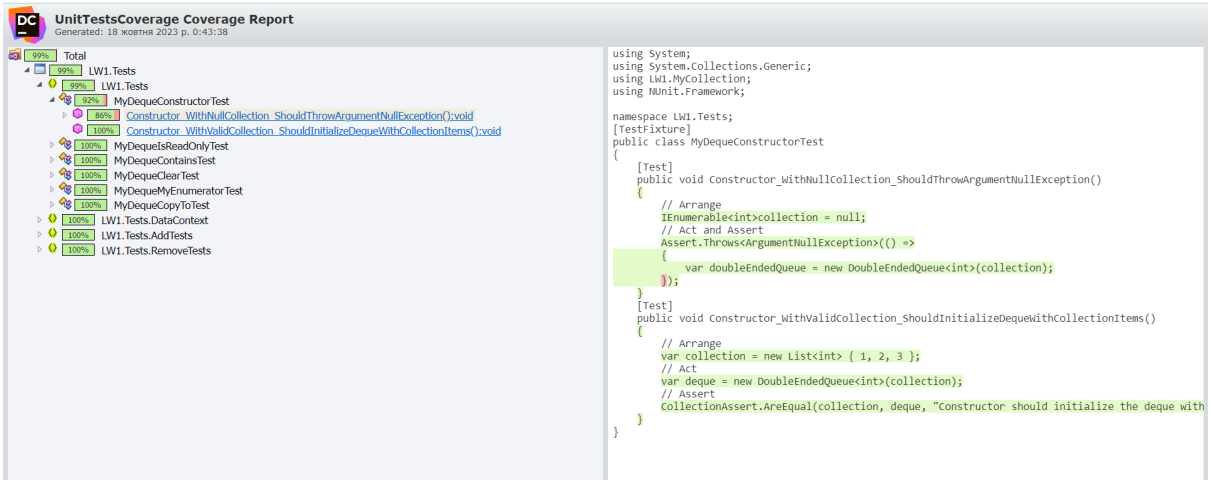

Рис.8 - Покриття коду (веб-версія)