Министерство науки и высшего образования Российской Федерации
Севастопольский государственный университет
Кафедра ИС

Отчет
по лабораторной работе №5
«Исследование способов модульного тестирования программного обеспечения в среде NUnit»
по дисциплине
«ТЕСТИРОВАНИЕ ПО»

Выполнил студент группы ИС/б-17-2-о
Горбенко К. Н.
Проверил
Тлуховская Н.П.

Севастополь
2019

1 ЦЕЛЬ РАБОТЫ

Исследовать эффективность использования методологии TDD при разработке программного обеспечения. Получить практические навыки использования фреймворка NUnit для модульного тестирования программного обеспечения.

2 ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Реализовать на языке C один из классов, спроектированных в лабораторной работе № 1. Методы класса при этом не реализовывать.

2. Разработать для созданного класса набор модульных тестов, включающий тесты для каждого метода.

3. Запустить набор тестов, проанализировать и сохранить результаты.

4. Поочередно реализовать методы класса, выполняя тестирование при каждом изменении программного кода.

5. После того, как весь набор тестов будет выполняться успешно, реализацию классов можно считать завершенной.

3 ХОД РАБОТЫ

Для тестирования выберем класс `MatrixOperations`:

```
1 public static class MatrixOperations
2     {
3         public static IEnumerable<int> GetSumsOfColumns(int[,] matrix)
4         {
5             throw new NotImplementedException();
6         }
7     }
```

Класс содержит один метод, в обязанности которого входит вычисление сумм положительных элементов в столбцах матрицы. При этом, если передать в метод неквадрутную матрицу, то происходит ошибка.

Создадим тестовые случаи:

1. Матрица - null.

Ожидаемый результат - ошибка.

2. Передается неквадратная матрица.

Ожидаемый результат - ошибка.

3. Матрица состоит из одного положительного элемента.

Ожидаемый результат - массив с одним элементом.

4. Матрица состоит из одного отрицательного элемента.

Ожидаемый результат - массив с одним элементом (0).

5. Квадратная матрица без отрицательных элементов.

Ожидаемый результат - массив с суммами всех элементов по столбцам.

6. Квадратная матрица с отрицательными элементами.

Ожидаемый результат - массив с суммами всех элементов по столбцам без учета отрицательных.

Опишем тесты:

```
1  public class MatrixOperationsTests
2  {
3      [Test]
4      public void PassingNullIssuesException()
5      {
6          Assert.Throws<ArgumentNullException>(() =>
               GetSumsOfPositiveElementsInColumns(null));
7      }
8
9      [Test]
10     public void PassingNonSquareMatrixIssuesException()
11     {
12         var matrix = new[,] {{1, 3, 5}, {2, 3, 4}};
13
14         Assert.Throws<ArgumentException>(() =>
               GetSumsOfPositiveElementsInColumns(matrix));
15     }
16
17     [Test]
18     public void SingleNegativeItem()
19     {
20         var matrix = new[,] {{-15}};
21         var expected = new[]{0};
22
23         var actual = GetSumsOfPositiveElementsInColumns(matrix);
24
25         Assert.That(expected, Is.EqualTo(actual));
26     }
27
28     [Test]
29     public void SinglePositiveItem()
30     {
31         var matrix = new[,] {{3}};
32         var expected = new[] {3};
```

```
33
34          var actual = GetSumsOfPositiveElementsInColumns(matrix);
35
36          Assert.That(expected, Is.EqualTo(actual));
37      }
38
39      [Test]
40      public void SquareMatrixWithoutNegativeElements()
41      {
42          var matrix = new[,]
43          {
44              {15, 10, 36},
45              {13, 0, 14},
46              {62, 15, 1235}
47          };
48          var expected = new[] {90, 25, 1285};
49
50          var actual = GetSumsOfPositiveElementsInColumns(matrix);
51
52          Assert.That(expected, Is.EqualTo(actual));
53      }
54
55      [Test]
56      public void SquareMatrixWithNegativeElements()
57      {
58          var matrix = new[,]
59          {
60              {-30, 10, 36},
61              {13, 0, 14},
62              {62, -5, 1235}
63          };
64          var expected = new[] {75, 10, 1285};
65
66          var actual = GetSumsOfPositiveElementsInColumns(matrix);
67
68          Assert.That(expected, Is.EqualTo(actual));
69      }
70  }
```

Реализуем класс:

```
1  public static class MatrixOperations
2  {
3      public static IEnumerable<int> GetSumsOfPositiveElementsInColumns(int[,]
           matrix)
4      {
5          if (matrix == null)
6          {
7              throw new ArgumentNullException(nameof(matrix));
```

```
 8          }
 9
10          if (matrix.GetLength(0) != matrix.GetLength(1))
11          {
12              throw new ArgumentException("Only square martixes are allowed.");
13          }
14
15          return matrix
16                  .GetColumns()
17                  .Select(column => column.Where(x => x > 0).Sum());
18      }
19  }
```

ВЫВОДЫ

В ходе лабораторной работы было выполнено тестирование класса с помощью фреймворка NUnit с использованием подхода TDD. TDD предполагает обдумывание реализации модулей и написание тестов к ним перед написанием самих модулей. Это позволяет описать функционал модуля таким образом, чтобы при дальнейшей его модификации не допустить таких изменений, которые повлияют на основной функционал.

Кроме того, модуль организовывается таким образом, чтобы удовлетворять лишь требованиям, возлагаемым на низх тестами. Таким образом, меньше времени тратится непосредственно на разработку, так как считается, что программист обдумал детали реализации еще при написании тестов.