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

Отчет
по лабораторной работе №1
«Исследование способов анализа областей эквивалентности и построения тестовых последовательностей»
по дисциплине
«ТЕСТИОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ»

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

Севастополь
2019

1 ЦЕЛЬ РАБОТЫ

Исследовать способы анализа областей эквивалентности входных данных для тестирования программного обеспечения. Приобрести практические навыки составления и построения тестовых последовательностей.

2 ЗАДАНИЕ НА РАБОТУ

Для **варианта № 5** заданы следующие требования к программам:

1. Дана целочисленная квадратная матрица. Определить сумму элементов в тех столбцах, которые не содержат отрицательных элементов.

2. Дана строка. Удалить в данной строке символ, стоящий на заданной позиции.

3. Программа, которая считывает текст из файла и выводит его на экран, меняя местами каждые два соседних слова.

Для каждой из программ необходимо:

1. Написать программу, выполняющую заданные действия.
2. Определить области эквивалентности входных данных.
3. Составить примеры тестовых последовательностей.

3 ТЕКСТ ПРОГРАММНЫХ МОДУЛЕЙ

3.1 Операции над матрицами

Была написана следующая программа:

```
1 public static class MatrixOperations
2 {
3     public static IEnumerable<int> GetSumOfColumnsWithoutNegativeElements(int
          [,] matrix)
4     {
5         if (matrix == null) throw new ArgumentNullException($"{nameof(matrix)}
              instance was null");
6         if (matrix.GetLength(0) != matrix.GetLength(1)) throw new
              ArgumentException($"Only square martixes are allowed");
7
8         return matrix
9                 .GetColumns()
10                .Where(column => column.All(items => items >= 0))
```

```
11                  .Select(column => column.Sum());
12      }
13  }
14
15  public static IEnumerable<int[]> GetColumns(this int[,] array)
16  {
17      if (array == null) throw new ArgumentNullException($"{nameof(array)}
            instance was null");
18
19      for (var j = 0; j < array.GetLength(1); j++)
20      {
21          var column = new int[array.GetLength(0)];
22
23          for (var i = 0; i < array.GetLength(0); i++)
24              column[i] = array[i, j];
25
26          yield return column;
27      }
28  }
```

3.2 Операции над строками

Была написана следующая программа:

```
1  public static class StringOperations
2  {
3      public static string RemoveAt(string source, int position) => source.Remove
           (position, 1);
4  }
```

3.3 Операции над текстом

```
1  public class TextOperations
2  {
3      private readonly StreamReader streamReader;
4      private readonly string[] delimiters = { " ", ".", ",", "?", "!", "(", ")",
           ":", ";", Environment.NewLine };
5
6      public TextOperations(StreamReader streamReader)
7          => this.streamReader = streamReader ?? throw new ArgumentNullException(
               $"{nameof(streamReader)} instance was null");
8
9      public string ReverseEveryTwoWords()
10     {
11         var source = streamReader.ReadToEnd();
12
13         return string.Join(" ", source
```

```
14                                          .Split(delimiters, StringSplitOptions.
                                                RemoveEmptyEntries)
15                                          .Select((word, i) => new { Value = word,
                                                Index = i })
16                                          .GroupBy(x => x.Index / 2)
17                                          .Select(group => group.Select(x => x.Value)
                                                .Reverse())
18                                          .SelectMany(x => x)
19              );
20      }
21 }
```

4 СОСТАВЛЕНИЕ ОБЛАСТЕЙ ЭКВИВАЛЕНТНОСТИ И ТЕСТОВЫХ ПРИМЕРОВ

4.1 Тестирование программы № 1

Определим области эквивалентности.

1. По размеру матрицы:
    a. Матрица состоит из одного элемента.
    b. Матрица состоит из нескольких элементов.
2. По наличию и расположению отрицательных элементов:
    a. Отрицательные элементы отсутствуют.
    b. Существует один отрицательный элемент.
    c. Существует несколько отрицательных элементов.

Составим тестовые примеры:

1.a,2.a: [3];

1.a,2.b: [-3];

1.b,2.a:

$$\begin{pmatrix} 15, 10, 36 \\ 23, 0, 14 \\ 62, 15, 35 \end{pmatrix}$$

1.b,2.b:

$$\begin{pmatrix} 13, 2, 14, 16 \\ 0, 54, -2, 9 \\ 62, 0, 21, 15 \\ 64, 0, 11, 2 \end{pmatrix}$$

1.b,2.c:

$$\begin{pmatrix} 85, 76, -53, 16, \\ 0, 637, -2, -12 \\ -15, 0, 5, 90 \\ 64, 126, 11, 2 \end{pmatrix}$$

4.2 Тестирование программы № 2

Опредедим области эквивалентности:

1. По размеру строки:
    a. Пустая строка.
    b. Длина строки - 1.
    c. Длина строки > 1.
2. По позиции удаляемого символа:
    a. Удаляется символ за пределами множества индексов строки.
    b. Удаляется первый элемент строки.
    c. Удаляется последний элемент строки.
    d. удаляется средний элемент строки.

Составим тестовые примеры:

1.a: «», любой индекс.

1.b,2.a: «a», 1.

1.b,2.b: «b», 0.

1.b,2.c: «C», 0.

1.c,2.a: «The quick brown fox jumps over the lazy dog.», 44.

1.c,2.b: «documentclass[a4paper,14pt]extarticle», 0.

1.c,2.c: «Hello, here is some text without a meaning. This text should show what a

printed text will look like at this place. If you read this text, you will get no information.», 173.

1.c,2.d: «(var i = 0; i < array.GetLength(0); i++», 14.

4.3 Тестирование программы № 3

Опредедим области эквивалентности:

1. По количеству слов:
    a. Пустая строка
    b. Одно слово
    c. Четное количество слов
    d. Нечетное количество слов

Создадим тестовые примеры:

1.a: «».

1.b: «Высокопревосходительство».

1.c: «После того, как определены области эквивалентности, выбираются тестовые последовательности, принадлежащие каждой из областей. При этом руководствуются следующими правилами».

1.d: «ветвей это метод структурного тестирования, при котором проверяются все независимо выполняемые ветви компонента или программы. Если выполняются все независимые ветви, то и все операторы должны выполняться, по крайней мере».

5 ВЫВОДЫ

В ходе лабораторной работы было исследовано тестирование ПО, представляющего собой черный ящик. Для тестирования черного ящика необходимо разбить множество возможных входных данных программы на области эквивалентности, на которых, как предполагается, программа ведет себя одинаково. После этого создаются тестовые примеры, содержащие входные данные из разных областей.

При тестировании черного ящика крайне необходимо создавать тестовые примеры, содержащие различные комбинации разных видов областей входных

данных и их размера.