

**Московский государственный технический  
университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра ИУ5. Курс «Базовые компоненты интернет-технологий»

Отчет по лабораторной работе №4

Выполнила:		Проверил:
студент группы ИУ5-31		преподаватель каф. ИУ5
Абросимова Надежда		
Подпись и дата:		Подпись и дата:

г. Москва, 2017 г.

## **Задание**

Разработать программу, реализующую работу с файлами.

1. Программа должна быть разработана в виде приложения Windows Forms на языке C#. По желанию вместо Windows Forms возможно использование WPF (Windows Presentation Foundation).
2. Добавить кнопку, реализующую функцию чтения текстового файла в список слов List<string>.
3. Для выбора имени файла используется класс OpenFileDialog, который открывает диалоговое окно с выбором файла. Ограничить выбор только файлами с расширением «.txt».
4. Для чтения из файла рекомендуется использовать статический метод ReadAllText() класса File (пространство имен System.IO). Содержимое файла считывается методом ReadAllText() в виде одной строки, далее делится на слова с использованием метода Split() класса string. Слова сохраняются в список List<string>.
5. При сохранении слов в список List<string> дубликаты слов не записываются. Для проверки наличия слова в списке используется метод Contains().
6. Вычислить время загрузки и сохранения в список с использованием класса Stopwatch (пространство имен System.Diagnostics). Вычисленное время вывести на форму в поле ввода (TextBox) или надпись (Label).
7. Добавить на форму поле ввода для поиска слова и кнопку поиска. При нажатии на кнопку поиска осуществлять поиск введенного слова в списке. Слово считается найденным, если оно входит в элемент списка как подстрока (метод Contains() класса string).
8. Добавить на форму список (ListBox). Найденные слова выводить в список с использованием метода «название\_списка.Items.Add()». Вызовы метода «название\_списка.Items.Add()» должны находиться между вызовами методов «название\_списка.BeginUpdate()» и «название\_списка.EndUpdate()».
9. Вычислить время поиска с использованием класса Stopwatch. Вычисленное время вывести на форму в поле ввода (TextBox) или надпись (Label)

## **Текст программы**

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
```

```
using System.IO;
using System.Diagnostics;
```

```
namespace Лаб_4._1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        /// <summary>
        /// Список слов
        /// </summary>
        List<string> list = new List<string>();

        private void label1_Click(object sender, EventArgs e)
        {

        }

        private void label3_Click(object sender, EventArgs e)
        {

        }

        private void label5_Click(object sender, EventArgs e)
        {

        }

        private void buttonClose_Click_1(object sender, EventArgs e)
        {
            this.Close();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            OpenFileDialog fd = new OpenFileDialog();
            fd.Filter = "текстовые файлы|*.txt";
            if (fd.ShowDialog() == DialogResult.OK)
            {
                Stopwatch t = new Stopwatch();
            }
        }
    }
}
```

```

t.Start();
//Чтение файла в виде строки
string text = File.ReadAllText(fd.FileName);
//Разделительные символы для чтения из файла
char[] separators = new char[] { ' ', '.', ',', '!', '?', '/', '\t', '\n' };
string[] textArray = text.Split(separators);
foreach (string strTemp in textArray)
{
    //Удаление пробелов в начале и конце строки
    string str = strTemp.Trim();
    //Добавление строки в список, если строка не содержится в
списке
    if (!list.Contains(str)) list.Add(str);
}

t.Stop();
this.textBoxFileReadTime.Text = t.Elapsed.ToString();
this.textBoxFileReadCount.Text = list.Count.ToString();
}
else
{
    MessageBox.Show("Необходимо выбрать файл");
}
}

private void button2_Click(object sender, EventArgs e)
{
    //Слово для поиска
    string word = this.textBoxFind.Text.Trim();

    //Если слово для поиска не пусто
    if (!string.IsNullOrEmpty(word) && list.Count > 0)
    {
        //Слово для поиска в верхнем регистре
        string wordUpper = word.ToUpper();
        //Временные результаты поиска
        List<string> tempList = new List<string>();
        Stopwatch t = new Stopwatch();
        t.Start();
        foreach (string str in list)
        {
            if (str.ToUpper().Contains(wordUpper))
            {
                tempList.Add(str);
            }
        }
    }
}

```

```

    }
}
t.Stop();
this.textBoxExactTime.Text = t.Elapsed.ToString();
this.listBoxResult.BeginUpdate();
//Очистка списка
this.listBoxResult.Items.Clear();
//Вывод результатов поиска
foreach (string str in tempList)
{
    this.listBoxResult.Items.Add(str);
}
this.listBoxResult.EndUpdate();
}
else
{
    MessageBox.Show("Необходимо выбрать файл и ввести
слово для поиска");
}
}

```

```

private void button3_Click(object sender, EventArgs e)
{
    //Слово для поиска
    string word = this.textBoxFind.Text.Trim();

}

```

```

private void button4_Click(object sender, EventArgs e)
{
    //Имя файла отчета
    string TempReportFileName = "Report_" +
    DateTime.Now.ToString("dd_MM_yyyy_hhmmss");
    //Диалог сохранения файла отчета
    SaveFileDialog fd = new SaveFileDialog();
    fd.FileName = TempReportFileName;
    fd.DefaultExt = ".html";
    fd.Filter = "HTML Reports|*.html";
    if (fd.ShowDialog() == DialogResult.OK)
    {
        string ReportFileName = fd.FileName;
        //Формирование отчета
        StringBuilder b = new StringBuilder();
        b.AppendLine("<html>");
        b.AppendLine("<head>");
    }
}

```

```

        b.AppendLine("<meta http-equiv='Content-Type' content='text/html; charset = UTF - 8'/>");
        b.AppendLine("<title>" + "Отчет: " + ReportFileName + "</title>");
        b.AppendLine("</head>");
        b.AppendLine("<body>");
        b.AppendLine("<h1>" + "Отчет: " + ReportFileName + "</h1>");
        b.AppendLine("<table border='1'>");
        b.AppendLine("<tr>");
        b.AppendLine("<td>Время чтения из файла</td>");
        b.AppendLine("<td>" + this.textBoxFileReadTime.Text + "</td>");
        b.AppendLine("</tr>");
        b.AppendLine("<tr>");
        b.AppendLine("<td>Количество уникальных слов в файле</td>");
        b.AppendLine("<td>" + this.textBoxFileReadCount.Text + "</td>");
        b.AppendLine("</tr>");
        b.AppendLine("<tr>");
        b.AppendLine("<td>Слово для поиска</td>");
        b.AppendLine("<td>" + this.textBoxFind.Text + "</td>");
        b.AppendLine("</tr>");
        b.AppendLine("<tr>");
        b.AppendLine("<td>Максимальное расстояние для нечеткого поиска </ td >");
        b.AppendLine("<td>" + this.textBoxMaxDist.Text + "</td>");
        b.AppendLine("</tr>");
        b.AppendLine("<tr>");
        b.AppendLine("<td>Время четкого поиска</td>");
        b.AppendLine("<td>" + this.textBoxExactTime.Text + "</td>");
        b.AppendLine("</tr>");
        b.AppendLine("<tr>");
        b.AppendLine("<td>Время нечеткого поиска</td>");
        b.AppendLine("<td>" + this.textBoxApproxTime.Text + "</td>");
        b.AppendLine("</tr>");
        b.AppendLine("<tr valign='top'>");
        b.AppendLine("<td>Результаты поиска</td>");
        b.AppendLine("<td>");
        b.AppendLine("<ul>");
        foreach (var x in this.listBoxResult.Items)
        {
            b.AppendLine("<li>" + x.ToString() + "</li>");
        }
        b.AppendLine("</ul>");
        b.AppendLine("</td>");
    }

```

```

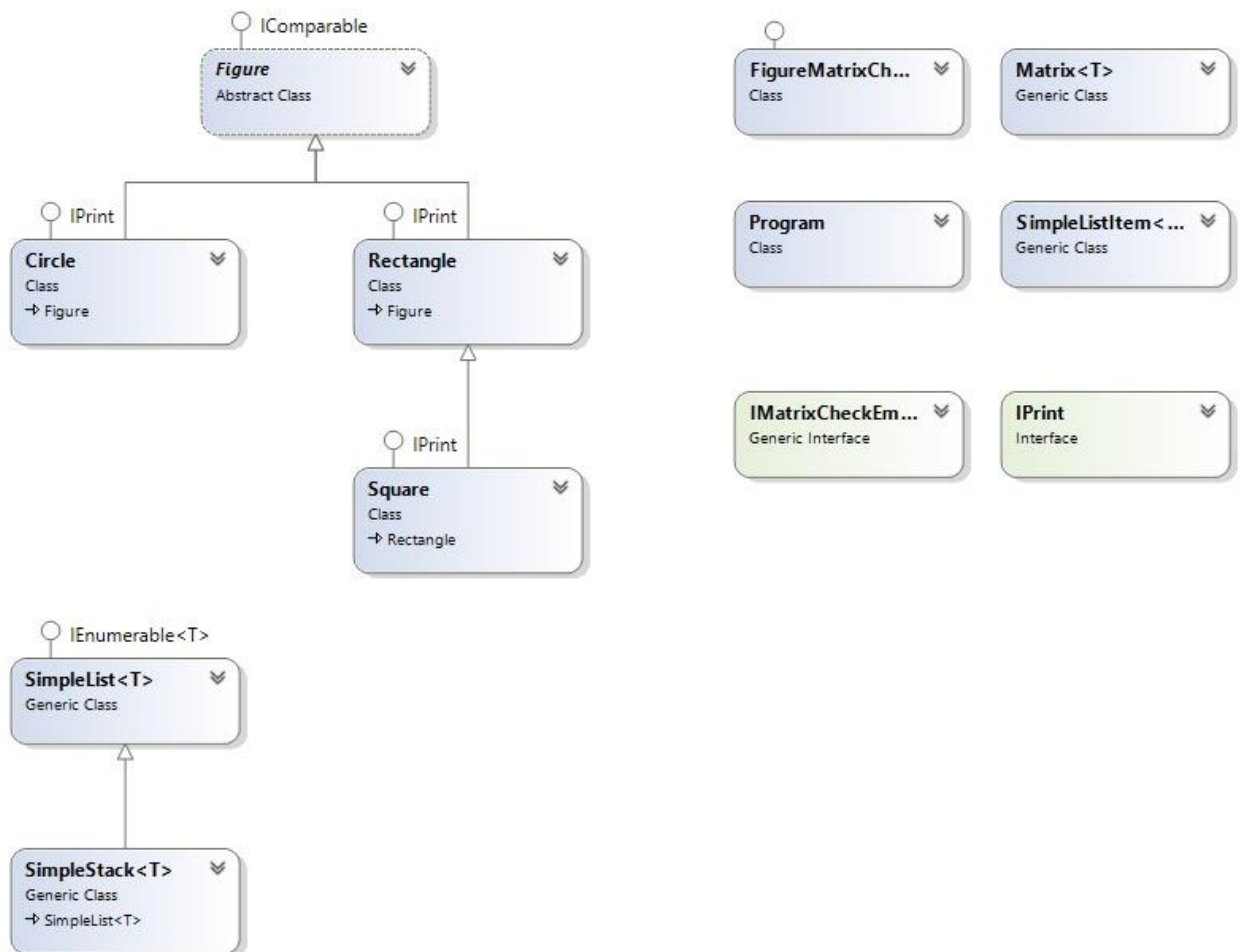
        b.AppendLine("</tr>");
        b.AppendLine("</table>");
        b.AppendLine("</body>");
        b.AppendLine("</html>");
        //Сохранение файла
        File.AppendAllText(ReportFileName, b.ToString());
        MessageBox.Show("Отчет сформирован. Файл: " +
ReportFileName);
    }

}

}

```

## Диаграмма классов



## Результат

Form1

Чтение из файла

Время чтения из файла:

Количество уникальных слов в файле:

Слово для поиска:

Чёткий поиск

Время чёткого поиска:

Максимальное расстояние для нечёткого поиска:

Количество потоков:

Вычисленное количество потоков:

Время нечёткого поиска:

Параллельный нечёткий поиск

Сохранение отчёта

Выход

Лаб\_4\_1.Form1

Необходимо выбрать файл

OK

```
{
}

private void label3_Click(object sender, EventArgs e)
{
}
}
```

Form1

Чтение из файла

Время чтения из файла: 00:00:00.0020453

Количество уникальных слов в файле: 524

Слово для поиска: текст

Чёткий поиск

Время чёткого поиска: 00:00:00.0001337

Максимальное расстояние для нечёткого поиска:

Количество потоков:

Вычисленное количество потоков:

Время нечёткого поиска:

Параллельный нечёткий поиск

текст  
тексте  
текста)  
тексту  
текста  
текста;

Сохранение отчёта

Выход