# Отчёт по лабораторной работе №4, 5
# по курсу
# «Базовые компоненты интернет-технологий»

Выполнила: студентка группы ИУ5-33

Нурлыева Дана

Москва, 2017

# Описание задания лабораторной работы

## Лабораторная работа №4

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.
2. Добавить кнопку, реализующую функцию чтения 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 находится 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).

**Лабораторная работа №5**

Разработать программу, реализующую вычисление расстояния Левенштейна с использованием алгоритма Вагнера-Фишера.

1. Программа должна быть разработана в виде библиотеки классов на языке C#.
2. Использовать самый простой вариант алгоритма без оптимизации.
3. Дополнительно возможно реализовать вычисление расстояния Дамерау-Левенштейна (с учетом перестановок соседних символов).
4. Модифицировать предыдущую лабораторную работу, вместо поиска подстроки используется вычисление расстояния Левенштейна.
5. Предусмотреть отдельное поле ввода для максимального расстояния. Если расстояние Левенштейна между двумя строками больше максимального, то строки считаются несовпадающими и не выводятся в список результатов.

# Текст программы на языке C#

### Form1.cs

```
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 Lab_4_5
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        /// <summary>
        /// Список слов
        /// </summary>
        List<string> list = new List<string>();
        private void buttonLoadFile_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 buttonExit_Click_1(object sender, EventArgs e)
        {
            this.Close();
            //Application.Exit();
        }

        private void buttonLoadFile_Click_1(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 buttonExact_Click_1(object sender, EventArgs e)
        {
            //Слово для поиска
            string word = this.textBoxFind.Text.Trim();
            //Если слово для поиска не пусто
            if (!string.IsNullOrWhiteSpace(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 buttonApprox_Click_1(object sender, EventArgs e)
        {
            //Слово для поиска
            string word = this.textBoxFind.Text.Trim();
            //Если слово для поиска не пусто
            if (!string.IsNullOrWhiteSpace(word) && list.Count > 0)
            {
                int maxDist;
                if (!int.TryParse(this.textBoxMaxDist.Text.Trim(), out
maxDist))
                {
                    MessageBox.Show("Необходимо указать максимальное
расстояние");
                    return;
                }
                if (maxDist < 1 || maxDist > 5)
                {
                    MessageBox.Show("Максимальное расстояние должно быть в
диапазоне от 1 до 5");
                    return;
                }

                //Слово для поиска в верхнем регистре
                string wordUpper = word.ToUpper();
```

```
                //Временные результаты поиска
                List<Tuple<string, int>> tempList = new List<Tuple<string,
int>>();
                Stopwatch t = new Stopwatch();
                t.Start();

                foreach (string str in list)
                {
                    //Вычисление расстояния Дамерау-Левенштейна
                    int dist = EditDistance.Distance(str.ToUpper(), wordUpper);
                    //Если расстояние меньше порогового, то слово добавляется в
результат
                    if (dist <= maxDist)
                    {
                        tempList.Add(new Tuple<string, int>(str, dist));
                    }
                }

                t.Stop();
                this.textBoxApproxTime.Text = t.Elapsed.ToString();

                this.listBoxResult.BeginUpdate();

                //Очистка списка
                this.listBoxResult.Items.Clear();

                //Вывод результатов поиска
                foreach (var x in tempList)
                {
                    string temp = x.Item1 + "(расстояние=" + x.Item2.ToString()
+ ")";
                    this.listBoxResult.Items.Add(temp);
                }
                this.listBoxResult.EndUpdate();
            }
            else
            {
                MessageBox.Show("Необходимо выбрать файл и ввести слово для
поиска");
            }
        }

        private void buttonSaveReport_Click_1(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);
                }
            }

            private void listBoxResult_SelectedIndexChanged(object sender,
       EventArgs e)
            {

            }
        }
       }
```

**EditDistance.cs**

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lab_4_5
{
    public static class EditDistance
    {
        /// <summary>
        /// Вычисление расстояния Дамерау-Левенштейна
        /// </summary>
        public static int Distance(string str1Param, string str2Param)
        {
            if ((str1Param == null) || (str2Param == null)) return -1;

            int str1Len = str1Param.Length;
            int str2Len = str2Param.Length;

            //Если хотя бы одна строка пустая, возвращается длина другой строки
            if ((str1Len == 0) && (str2Len == 0)) return 0;
            if (str1Len == 0) return str2Len;
            if (str2Len == 0) return str1Len;

            //Приведение строк к верхнему регистру
            string str1 = str1Param.ToUpper();
            string str2 = str2Param.ToUpper();

            //Объявление матрицы
            int[,] matrix = new int[str1Len + 1, str2Len + 1];

            //Инициализация нулевой строки и нулевого столбца матрицы
            for (int i = 0; i <= str1Len; i++) matrix[i, 0] = i;
```

```
            for (int j = 0; j <= str2Len; j++) matrix[0, j] = j;

            //Вычисление расстояния Дамерау-Левенштейна
            for (int i = 1; i <= str1Len; i++)
            {
                for (int j = 1; j <= str2Len; j++)
                {
                    //Эквивалентность символов, переменная symbEqual соответствует
m(s1[i],s2[j])
                    int symbEqual = ((str1.Substring(i - 1, 1) == str2.Substring(j -
1, 1)) ? 0 : 1);

                    int ins = matrix[i, j - 1] + 1; //Добавление
                    int del = matrix[i - 1, j] + 1; //Удаление
                    int subst = matrix[i - 1, j - 1] + symbEqual; //Замена

                    //Элемент матрицы вычисляется как минимальный из трех случаев
                    matrix[i, j] = Math.Min(Math.Min(ins, del), subst);

                    //Дополнение Дамерау по перестановке соседних символов
                    if ((i > 1) && (j > 1) &&
                        (str1.Substring(i - 1, 1) == str2.Substring(j - 2, 1)) &&
                        (str1.Substring(i - 2, 1) == str2.Substring(j - 1, 1)))
                    {
                        matrix[i, j] = Math.Min(matrix[i, j], matrix[i - 2, j - 2] +
symbEqual);
                    }
                }
            }
            //Возвращается нижний правый элемент матрицы
            return matrix[str1Len, str2Len];
        }
    }
}
```

## Form1.Designer.cs

```
namespace Lab_4_5
{
    partial class Form1
    {
        /// <summary>
        /// Обязательная переменная конструктора.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Освободить все используемые ресурсы.
        /// </summary>
        /// <param name="disposing">истинно, если управляемый ресурс должен
быть удален; иначе ложно.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Код, автоматически созданный конструктором форм Windows
```

```
        /// <summary>
        /// Требуемый метод для поддержки конструктора — не изменяйте
        /// содержимое этого метода с помощью редактора кода.
        /// </summary>
        private void InitializeComponent()
        {
            this.textBoxFileReadTime = new System.Windows.Forms.TextBox();
            this.textBoxMaxDist = new System.Windows.Forms.TextBox();
            this.textBoxFind = new System.Windows.Forms.TextBox();
            this.textBoxExactTime = new System.Windows.Forms.TextBox();
            this.textBoxFileReadCount = new System.Windows.Forms.TextBox();
            this.textBoxApproxTime = new System.Windows.Forms.TextBox();
            this.listBoxResult = new System.Windows.Forms.ListBox();
            this.label1 = new System.Windows.Forms.Label();
            this.label2 = new System.Windows.Forms.Label();
            this.label3 = new System.Windows.Forms.Label();
            this.label4 = new System.Windows.Forms.Label();
            this.label5 = new System.Windows.Forms.Label();
            this.label6 = new System.Windows.Forms.Label();
            this.buttonLoadFile = new System.Windows.Forms.Button();
            this.buttonExact = new System.Windows.Forms.Button();
            this.buttonApprox = new System.Windows.Forms.Button();
            this.buttonSaveReport = new System.Windows.Forms.Button();
            this.buttonExit = new System.Windows.Forms.Button();
            this.SuspendLayout();
            //
            // textBoxFileReadTime
            //
            this.textBoxFileReadTime.Location = new System.Drawing.Point(312,
29);
            this.textBoxFileReadTime.Name = "textBoxFileReadTime";
            this.textBoxFileReadTime.Size = new System.Drawing.Size(100, 20);
            this.textBoxFileReadTime.TabIndex = 0;
            //
            // textBoxMaxDist
            //
            this.textBoxMaxDist.Location = new System.Drawing.Point(312, 144);
            this.textBoxMaxDist.Name = "textBoxMaxDist";
            this.textBoxMaxDist.Size = new System.Drawing.Size(100, 20);
            this.textBoxMaxDist.TabIndex = 1;
            //
            // textBoxFind
            //
            this.textBoxFind.Location = new System.Drawing.Point(206, 90);
            this.textBoxFind.Name = "textBoxFind";
            this.textBoxFind.Size = new System.Drawing.Size(100, 20);
            this.textBoxFind.TabIndex = 2;
            //
            // textBoxExactTime
            //
            this.textBoxExactTime.Location = new System.Drawing.Point(312,
118);
            this.textBoxExactTime.Name = "textBoxExactTime";
            this.textBoxExactTime.Size = new System.Drawing.Size(100, 20);
            this.textBoxExactTime.TabIndex = 3;
            //
```

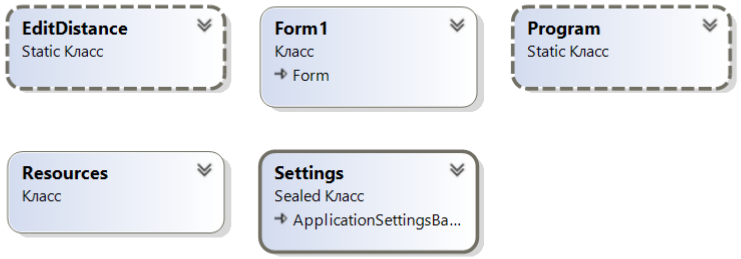
```
            // textBoxFileReadCount
            //
            this.textBoxFileReadCount.Location = new System.Drawing.Point(312,
55);
            this.textBoxFileReadCount.Name = "textBoxFileReadCount";
            this.textBoxFileReadCount.Size = new System.Drawing.Size(100, 20);
            this.textBoxFileReadCount.TabIndex = 4;
            //
            // textBoxApproxTime
            //
            this.textBoxApproxTime.Location = new System.Drawing.Point(312,
170);
            this.textBoxApproxTime.Name = "textBoxApproxTime";
            this.textBoxApproxTime.Size = new System.Drawing.Size(100, 20);
            this.textBoxApproxTime.TabIndex = 5;
            //
            // listBoxResult
            //
            this.listBoxResult.FormattingEnabled = true;
            this.listBoxResult.Location = new System.Drawing.Point(31, 222);
            this.listBoxResult.Name = "listBoxResult";
            this.listBoxResult.Size = new System.Drawing.Size(391, 108);
            this.listBoxResult.TabIndex = 6;
            this.listBoxResult.SelectedIndexChanged += new
System.EventHandler(this.listBoxResult_SelectedIndexChanged);
            //
            // label1
            //
            this.label1.AutoSize = true;
            this.label1.Location = new System.Drawing.Point(179, 32);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(127, 13);
            this.label1.TabIndex = 7;
            this.label1.Text = "Время чтения из файла";
            //
            // label2
            //
            this.label2.AutoSize = true;
            this.label2.Location = new System.Drawing.Point(106, 58);
            this.label2.Name = "label2";
            this.label2.Size = new System.Drawing.Size(200, 13);
            this.label2.TabIndex = 8;
            this.label2.Text = "Количество уникальных слов в файле";
            //
            // label3
            //
            this.label3.AutoSize = true;
            this.label3.Location = new System.Drawing.Point(102, 90);
            this.label3.Name = "label3";
            this.label3.Size = new System.Drawing.Size(98, 13);
            this.label3.TabIndex = 9;
            this.label3.Text = "Слово для поиска";
            //
            // label4
            //
            this.label4.AutoSize = true;
            this.label4.Location = new System.Drawing.Point(185, 121);
```

```
            this.label4.Name = "label4";
            this.label4.Size = new System.Drawing.Size(121, 13);
            this.label4.TabIndex = 10;
            this.label4.Text = "Время чёткого поиска";
            //
            // label5
            //
            this.label5.AutoSize = true;
            this.label5.Location = new System.Drawing.Point(46, 147);
            this.label5.Name = "label5";
            this.label5.Size = new System.Drawing.Size(260, 13);
            this.label5.TabIndex = 11;
            this.label5.Text = "Максимальное расстояние для нечёткого поиска";
            //
            // label6
            //
            this.label6.AutoSize = true;
            this.label6.Location = new System.Drawing.Point(173, 173);
            this.label6.Name = "label6";
            this.label6.Size = new System.Drawing.Size(133, 13);
            this.label6.TabIndex = 12;
            this.label6.Text = "Время нечёткого поиска";
            //
            // buttonLoadFile
            //
            this.buttonLoadFile.Location = new System.Drawing.Point(21, 25);
            this.buttonLoadFile.Name = "buttonLoadFile";
            this.buttonLoadFile.Size = new System.Drawing.Size(124, 24);
            this.buttonLoadFile.TabIndex = 13;
            this.buttonLoadFile.Text = "Чтение из файла";
            this.buttonLoadFile.UseVisualStyleBackColor = true;
            this.buttonLoadFile.Click += new
System.EventHandler(this.buttonLoadFile_Click_1);
            //
            // buttonExact
            //
            this.buttonExact.Location = new System.Drawing.Point(21, 116);
            this.buttonExact.Name = "buttonExact";
            this.buttonExact.Size = new System.Drawing.Size(124, 23);
            this.buttonExact.TabIndex = 14;
            this.buttonExact.Text = "Чёткий поиск";
            this.buttonExact.UseVisualStyleBackColor = true;
            this.buttonExact.Click += new
System.EventHandler(this.buttonExact_Click_1);
            //
            // buttonApprox
            //
            this.buttonApprox.Location = new System.Drawing.Point(21, 170);
            this.buttonApprox.Name = "buttonApprox";
            this.buttonApprox.Size = new System.Drawing.Size(123, 43);
            this.buttonApprox.TabIndex = 15;
            this.buttonApprox.Text = "Параллельный нечёткий поиск";
            this.buttonApprox.UseVisualStyleBackColor = true;
            this.buttonApprox.Click += new
System.EventHandler(this.buttonApprox_Click_1);
            //
            // buttonSaveReport
```

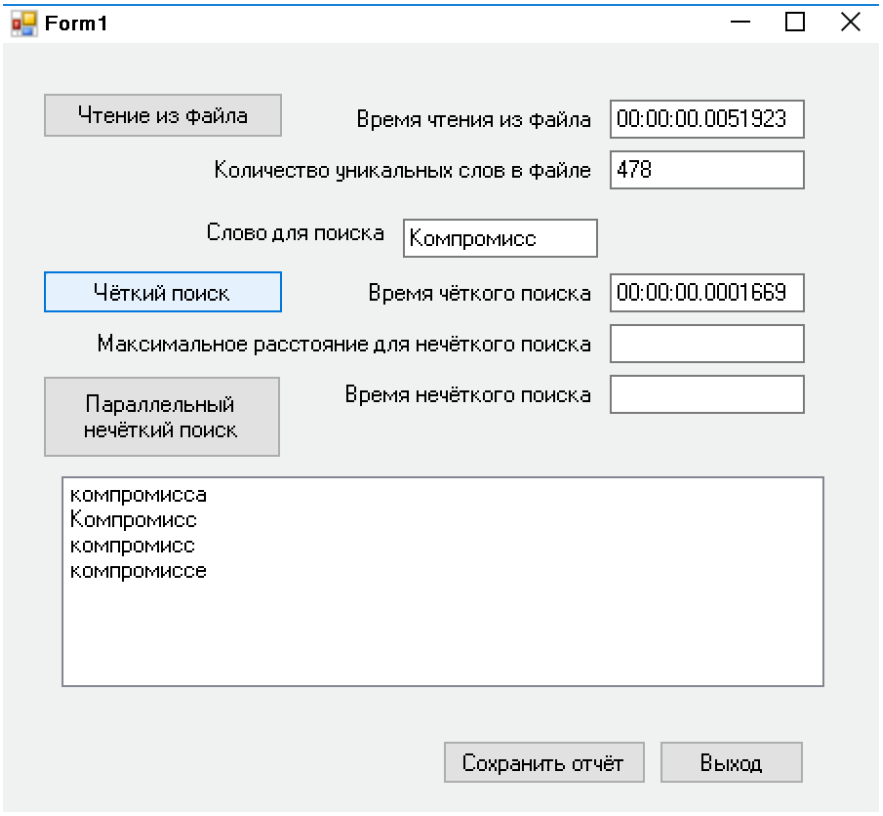
```
            //
            this.buttonSaveReport.Location = new System.Drawing.Point(226,
357);
            this.buttonSaveReport.Name = "buttonSaveReport";
            this.buttonSaveReport.Size = new System.Drawing.Size(105, 23);
            this.buttonSaveReport.TabIndex = 16;
            this.buttonSaveReport.Text = "Сохранить отчёт";
            this.buttonSaveReport.UseVisualStyleBackColor = true;
            this.buttonSaveReport.Click += new
System.EventHandler(this.buttonSaveReport_Click_1);
            //
            // buttonExit
            //
            this.buttonExit.Location = new System.Drawing.Point(337, 357);
            this.buttonExit.Name = "buttonExit";
            this.buttonExit.Size = new System.Drawing.Size(75, 23);
            this.buttonExit.TabIndex = 17;
            this.buttonExit.Text = "Выход";
            this.buttonExit.UseVisualStyleBackColor = true;
            this.buttonExit.Click += new
System.EventHandler(this.buttonExit_Click_1);
            //
            // Form1
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(450, 395);
            this.Controls.Add(this.buttonExit);
            this.Controls.Add(this.buttonSaveReport);
            this.Controls.Add(this.buttonApprox);
            this.Controls.Add(this.buttonExact);
            this.Controls.Add(this.buttonLoadFile);
            this.Controls.Add(this.label6);
            this.Controls.Add(this.label5);
            this.Controls.Add(this.label4);
            this.Controls.Add(this.label3);
            this.Controls.Add(this.label2);
            this.Controls.Add(this.label1);
            this.Controls.Add(this.listBoxResult);
            this.Controls.Add(this.textBoxApproxTime);
            this.Controls.Add(this.textBoxFileReadCount);
            this.Controls.Add(this.textBoxExactTime);
            this.Controls.Add(this.textBoxFind);
            this.Controls.Add(this.textBoxMaxDist);
            this.Controls.Add(this.textBoxFileReadTime);
            this.Name = "Form1";
            this.Text = "Form1";
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.TextBox textBoxFileReadTime;
        private System.Windows.Forms.TextBox textBoxMaxDist;
        private System.Windows.Forms.TextBox textBoxFind;
```

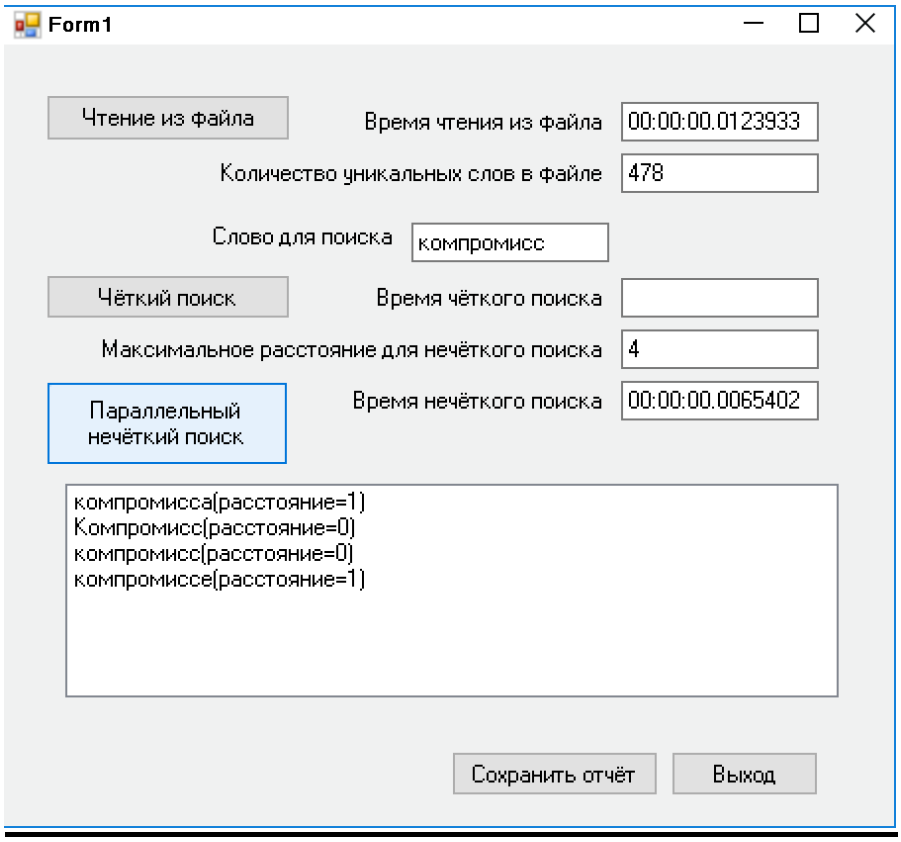
```
        private System.Windows.Forms.TextBox textBoxExactTime;
        private System.Windows.Forms.TextBox textBoxFileReadCount;
        private System.Windows.Forms.TextBox textBoxApproxTime;
        private System.Windows.Forms.ListBox listBoxResult;
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.Label label3;
        private System.Windows.Forms.Label label4;
        private System.Windows.Forms.Label label5;
        private System.Windows.Forms.Label label6;
        private System.Windows.Forms.Button buttonLoadFile;
        private System.Windows.Forms.Button buttonExact;
        private System.Windows.Forms.Button buttonApprox;
        private System.Windows.Forms.Button buttonSaveReport;
        private System.Windows.Forms.Button buttonExit;
    }
}
```

# Диаграмма классов



# Результаты выполнения программы



# Отчет: F:\Базовые компоненты интернет технологий\ДЗ\Report_15_11_2017_042305.html

| | |
|---|---|
| Время чтения из файла | 00:00:00.0074798 |
| Количество уникальных слов в файле | 478 |
| Слово для поиска | компромисс |
| Максимальное расстояние для нечеткого поиска | |
| Время четкого поиска | 00:00:00.0001632 |
| Время нечеткого поиска | |
| Результаты поиска | • компромисса<br>• Компромисс<br>• компромисс<br>• компромиссе |

---

# Отчет: F:\Базовые компоненты интернет технологий\ДЗ\Report_15_11_2017_042841.html

| | |
|---|---|
| Время чтения из файла | 00:00:00.0055195 |
| Количество уникальных слов в файле | 478 |
| Слово для поиска | компромисс |
| Максимальное расстояние для нечеткого поиска | 4 |
| Время четкого поиска | |
| Время нечеткого поиска | 00:00:00.0055999 |
| Результаты поиска | • компромисса(расстояние=1)<br>• Компромисс(расстояние=0)<br>• компромисс(расстояние=0)<br>• компромиссе(расстояние=1) |

---