**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»

Отчет по лабораторной работе №5

«Вычисление расстояния Ливенштейна»

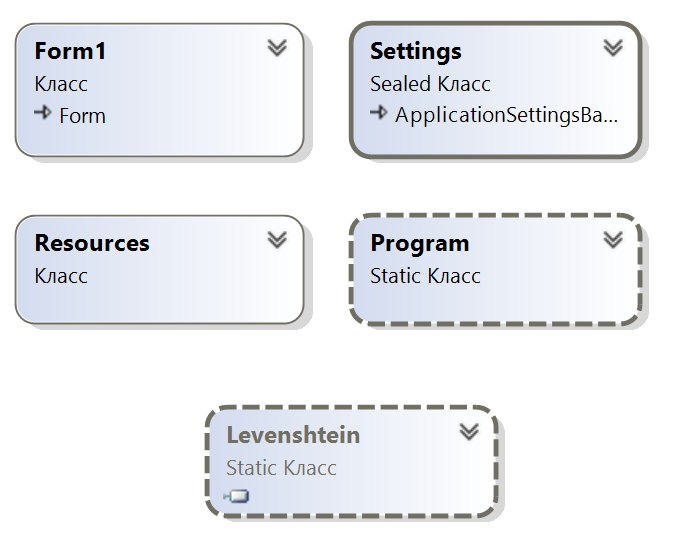
|  |  |  |
| --- | --- | --- |
| Выполнила: |  | Проверил: |
| студентка группы ИУ5-33 |  | преподаватель каф. ИУ5 |
| Ларичева М.В. |  | Гапанюк Ю.Е. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2018 г.

1. **Содержания задания**

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.
6. **Диаграмма классов**



1. **Текст программы**
   1. **Класс Levenshtein**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Levenshtein

{

public static class Levenshtein

{

public static int Distance(string str1, string str2)

{

int strlen1 = str1.Length, strlen2 = str2.Length;

//проверка на пустую строку

if (strlen1 == 0 || strlen2 == 0)

{

if (strlen1 != 0)

{

return strlen1;

}

else

{

return strlen2;

}

}

string s1 = str1.ToUpper(), s2 = str2.ToUpper();

int[,] matrix = new int[strlen1 + 1, strlen2 + 1];

//первая строка и столбец матрицы

for (int i = 0; i <= strlen1; i++)

{

matrix[i, 0] = i;

}

for (int j = 0; j <= strlen2; j++)

{

matrix[0, j] = j;

}

//остальные ячейки матрицы

for (int i = 1; i <= strlen1; i++)

{

for (int j = 1; j <= strlen2; j++)

{

int equal;

if (s1.Substring(i - 1, 1) == s2.Substring(j - 1, 1))

{

equal = 0;

}

else

{

equal = 1;

}

matrix[i, j] = Math.Min(Math.Min(matrix[i - 1, j] + 1, matrix[i, j - 1] + 1), matrix[i - 1, j - 1] + equal);

if ((i > 1) && (j > 1) &&

(s1.Substring(i - 1, 1) == s2.Substring(j - 1, 1)) &&

(s1.Substring(i - 2, 1) == s2.Substring(j - 2, 1)))

{

matrix[i, j] = Math.Min(matrix[i, j], matrix[i - 2, j - 2] + equal);

}

}

}

//возврат результата

return matrix[strlen1, strlen2];

}

public static void WriteDistance(string str1, string str2)

{

Console.WriteLine(Distance(str1, str2));

Console.ReadLine();

}

}

}

* 1. **Класс Form1**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.IO;

using System.Linq;

using System.Text;

using System.Diagnostics;

using System.Windows.Forms;

namespace Files {

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

/// <summary>

/// Список слов WordList, содержащихся в файле.

/// </summary>

public List<string> WordList { get; set; }

/// <summary>

/// Создание списка слов WordList, содержащихся в файле. Активация остальных элементов управления формы.

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void FileSelectionDialog\_FileOk(object sender, CancelEventArgs e)

{

//Сокращение полного имени файла для менее громоздкого отображения на экране

FileNameLabelShown.Text = "..." + FileSelectionDialog.FileName.Remove(0, FileSelectionDialog.FileName.LastIndexOf('\\') + 1);

Stopwatch sw = new Stopwatch();

sw.Start();

string FileContents = File.ReadAllText(FileSelectionDialog.FileName, Encoding.GetEncoding(1251));

string[] FileWords = FileContents.Split(' ', '\n', '\r', '!', ',', '?', '—', '.', '-', ':', ';');

List<string> list = new List<string>();

foreach (string s in FileWords)

{

if (!list.Contains(s) && s != "")

{

list.Add(s);

}

}

WordList = list;

sw.Stop();

ElapsedListCreationTime.Text = sw.ElapsedTicks.ToString() + " тиков";

WordSearchButton.Enabled = true;

WordSearchBox.Enabled = true;

WordListBox.Enabled = true;

WordListBoxClearButton.Enabled = true;

MaxDistanceBox.Enabled = true;

}

/// <summary>

/// Вызов диалога выбора текстового файла.

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void FileSelectionButton\_Click(object sender, EventArgs e)

{

FileSelectionDialog.ShowDialog();

}

/// <summary>

/// Поиск слова в списке WordList и вывод найденных слов в WordListBox.

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void WordSearchButton\_Click(object sender, EventArgs e)

{

//Обработка пользовательского ввода

//В случае ввода некорректных данных максимальное расстояние Левенштейна считается нулевым

if (!int.TryParse(MaxDistanceBox.Text, out int maxDist))

{

maxDist = 0;

MaxDistanceBox.Text = "0";

}

int foundWordsAmount = 0;

Stopwatch sw = new Stopwatch();

sw.Start();

for (int i = 0; i < WordList.Count; i++) {

if (Levenshtein.Levenshtein.Distance(WordSearchBox.Text, WordList[i]) <= maxDist)

{

if (!WordListBox.Items.Contains(WordList[i]))

{

WordListBox.BeginUpdate();

WordListBox.Items.Add(WordList[i]);

WordListBox.EndUpdate();

}

foundWordsAmount++;

}

}

sw.Stop();

ElapsedWordSearchTime.Text = sw.ElapsedTicks.ToString() + " тиков";

switch (foundWordsAmount)

{

case 0:

label1.Text = "Статус поиска: Слово не найдено";

break;

case 1:

label1.Text = "Статус поиска: Cлово найдено";

break;

default:

label1.Text = "Статус поиска: Найдено несколько слов";

break;

}

}

/// <summary>

/// Реализация функционала клавиши Enter при вводе слова для поиска в окне WordSearchBox

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void WordSearchBox\_Enter(object sender, EventArgs e)

{

AcceptButton = WordSearchButton;

}

/// <summary>

/// Предотвращение некорректной работы клавиши Enter при неактивном окне WordSearchBox

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void WordSearchBox\_Leave(object sender, EventArgs e)

{

AcceptButton = null;

}

/// <summary>

/// Возврат состояния программы к исходному.

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void WordListBoxClearButton\_Click(object sender, EventArgs e)

{

WordListBox.Items.Clear();

WordList.Clear();

FileNameLabelShown.Text = "Нет";

WordSearchButton.Enabled = false;

WordSearchBox.Enabled = false;

WordListBox.Enabled = false;

WordListBoxClearButton.Enabled = false;

}

}

}

* 1. **Класс Program**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Files

{

static class Program

{

/// <summary>

/// Главная точка входа для приложения.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}

1. **Экранные формы с примерами выполнения программы**

