**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра ИУ5. Курс «Базовые компоненты интернет-технологий»

Отчет по лабораторной работе №4 и №5

**«**Работа с файлами и расстояние Левеншнтейна»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-25б |  | доцент каф. ИУ5 |
| Шкарин Е.А. |  | Гапанюк Ю.Е. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2020 г.

# Постановка задачи:

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. 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. 5. При сохранении слов в список List 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).

# Текст программы:

//LevDist.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LevDist

{

public class Lev\_Dist

{

//private string word1;

//private string word2;

public Lev\_Dist()

{

}

static int Minimum(int a, int b, int c) => (a = a < b ? a : b) < c ? a : c;

static public int LevenshteinDistance(string word1, string word2)

{

var n = word1.Length + 1;

var m = word2.Length + 1;

var matrixD = new int[n, m];

const int deletionCost = 1;

const int insertionCost = 1;

for (var i = 0; i < n; i++)

{

matrixD[i, 0] = i;

}

for (var j = 0; j < m; j++)

{

matrixD[0, j] = j;

}

for (var i = 1; i < n; i++)

{

for (var j = 1; j < m; j++)

{

var substitutionCost = word1[i - 1] == word2[j - 1] ? 0 : 1;

matrixD[i, j] = Minimum(matrixD[i - 1, j] + deletionCost,

matrixD[i, j - 1] + insertionCost,

matrixD[i - 1, j - 1] + substitutionCost);

}

}

return matrixD[n - 1, m - 1];

}

}

}

//MainWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using System.IO;

using Microsoft.Win32;

using System.Diagnostics;

using System.Globalization;

using LevDist;

namespace wpf\_less1

{

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

private void open\_file\_Click(object sender, RoutedEventArgs e)

{

Stopwatch stopwatch = new Stopwatch();

stopwatch.Start();

output1.Clear();

output1.IsReadOnly = true;

OpenFileDialog dialog = new OpenFileDialog();

dialog.FileName = "Document";

dialog.DefaultExt = ".txt";

dialog.Filter = "Text documents (.txt)|\*.txt";

Nullable<bool> result = dialog.ShowDialog();

if (result == true)

{

using (StreamReader sr = new StreamReader(dialog.FileName))

{

string buf = sr.ReadToEnd();

string[] buf\_str = buf.Split();

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

for (int i = 0; i < buf\_str.Length; i++)

{

if(!string.IsNullOrWhiteSpace(buf\_str[i]))

list.Add(buf\_str[i]);

}

List<string> del = new List<string>(list.Intersect(buf\_str));

foreach(var item in del)

{

output1.Text += item + Environment.NewLine;

}

}

}

stopwatch.Stop();

time.Text = stopwatch.Elapsed.ToString();

}

private void create\_file\_Click(object sender, RoutedEventArgs e)

{

New\_file new\_file = new New\_file();

new\_file.Owner = this;

new\_file.Show();

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

Stopwatch stopwatch = new Stopwatch();

stopwatch.Start();

if (output1.Text.Length == 0)

{

MessageBox.Show("Откройте файл");

}

else

{

#region search

//List<string> search\_list= new List<string>();

//string buf = output1.Text;

//string str\_search = Search.Text;

//TextInfo myTI = new CultureInfo("ru-RU", false).TextInfo;

//str\_search = myTI.ToTitleCase(str\_search);

//if (buf.Contains(str\_search))

//{

// ListBox.Items.Add(Search.Text);

//}

//else

//{

// MessageBox.Show("Такого слова в этом файле нет");

//}

#endregion

Lev\_Dist lev = new Lev\_Dist();

string[] buf\_str = output1.Text.Split();

List<string> buf\_list = new List<string>();

for (int i = 0; i < buf\_str.Length; i++)

{

if (!string.IsNullOrWhiteSpace(buf\_str[i]))

{

buf\_list.Add(buf\_str[i]);

}

}

string str = Search.Text;

foreach (var item in buf\_list)

{

int dist = Lev\_Dist.LevenshteinDistance(str, item);

if (dist > 1)

{

continue;

}

else

{

ListBox.Items.Add(item);

using(var sr = new StreamWriter("Othchet.txt", true, Encoding.UTF8))

{

sr.WriteLine(item + "\n");

}

break;

}

}

}

Search.Clear();

stopwatch.Stop();

time.Text = stopwatch.Elapsed.ToString();

}

private void MenuItem\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

private void Lev\_Click\_1(object sender, RoutedEventArgs e)

{

int buf = 0;

if(!Int32.TryParse(Lev\_text.Text, out buf) || string.IsNullOrWhiteSpace(output1.Text))

{

MessageBox.Show("Вы ввели не число или не открыли файл, или не ввели само число");

}

else

{

Lev\_Dist lev = new Lev\_Dist();

string[] buf\_str = output1.Text.Split();

List <string> buf\_list= new List<string>();

for(int i = 0; i < buf\_str.Length; i++)

{

if (!string.IsNullOrWhiteSpace(buf\_str[i]))

{

buf\_list.Add(buf\_str[i]);

}

}

string str = Lev\_str.Text;

foreach (var item in buf\_list)

{

int dist = Lev\_Dist.LevenshteinDistance(str, item);

if (dist > buf)

{

continue;

}

else

{

Lev\_list.Items.Add(item);

}

}

Lev\_str.Clear();

}

}

}

}

//App.xaml.cs

using System;

using System.Collections.Generic;

using System.Configuration;

using System.Data;

using System.Linq;

using System.Threading.Tasks;

using System.Windows;

namespace wpf\_less1

{

/// <summary>

/// Логика взаимодействия для App.xaml

/// </summary>

public partial class App : Application

{

}

}

//New\_file.xaml.cs

using Microsoft.Win32;

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace wpf\_less1

{

public partial class New\_file : Window

{

public New\_file()

{

InitializeComponent();

}

private void Save\_button\_Click(object sender, RoutedEventArgs e)

{

SaveFileDialog saveFileDialog = new SaveFileDialog();

saveFileDialog.DefaultExt = "txt";

if (saveFileDialog.ShowDialog() == true)

File.WriteAllText(saveFileDialog.FileName, Save\_text.Text);

Save\_text.Clear();

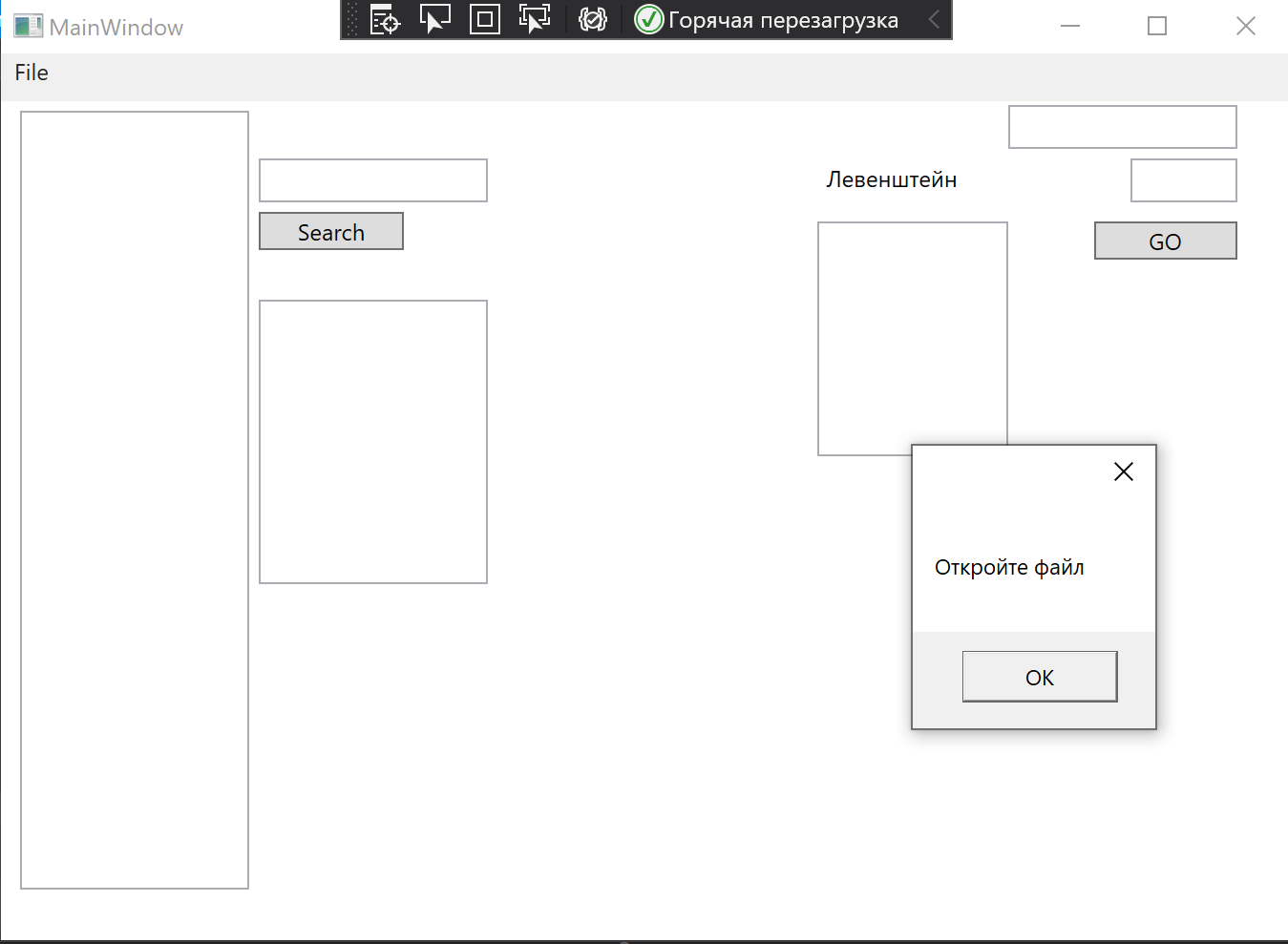
}

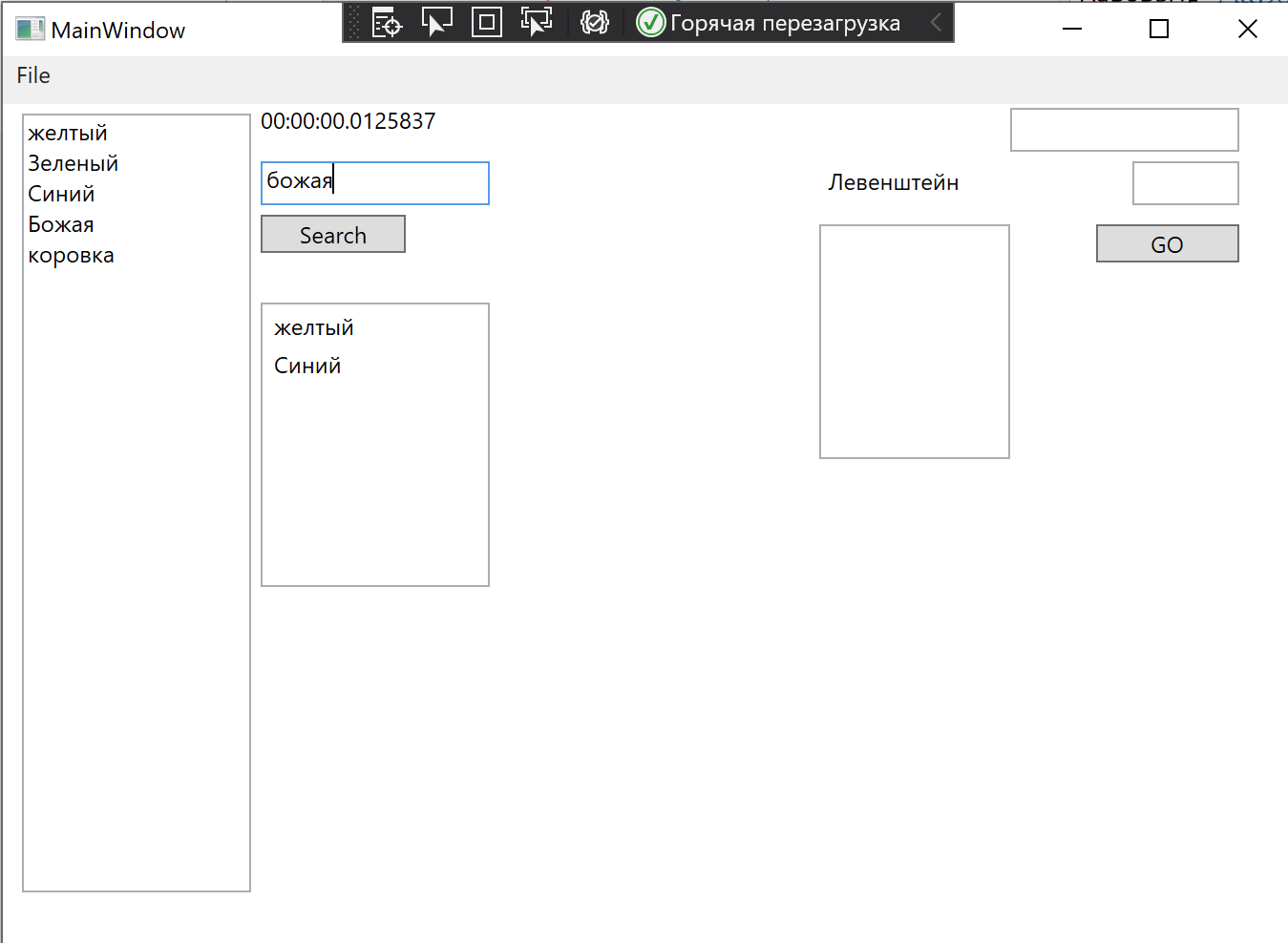
}

}

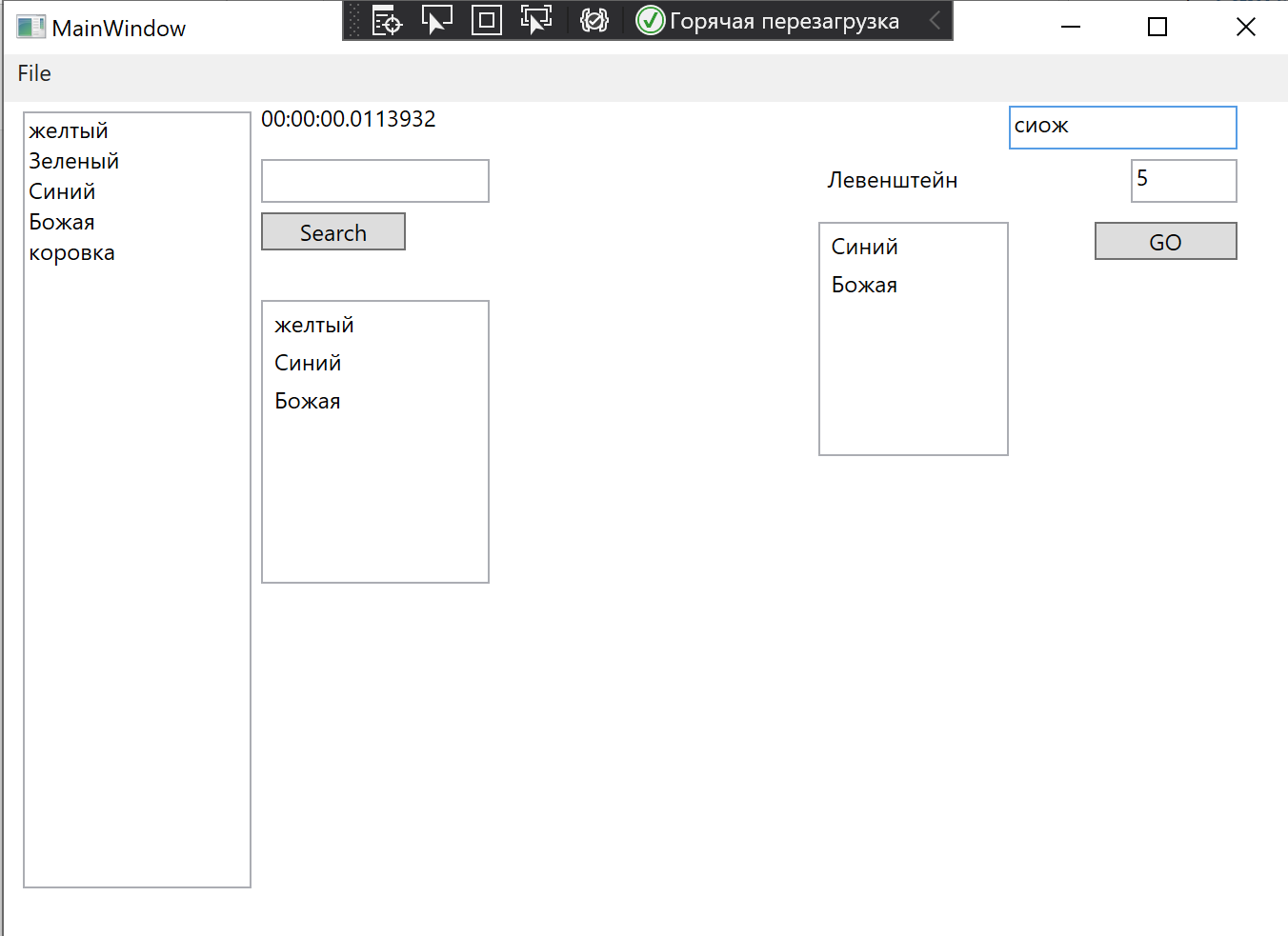
Тесты программы:

При попытке поиска без открытого файла, выскакивает окно ошибки.

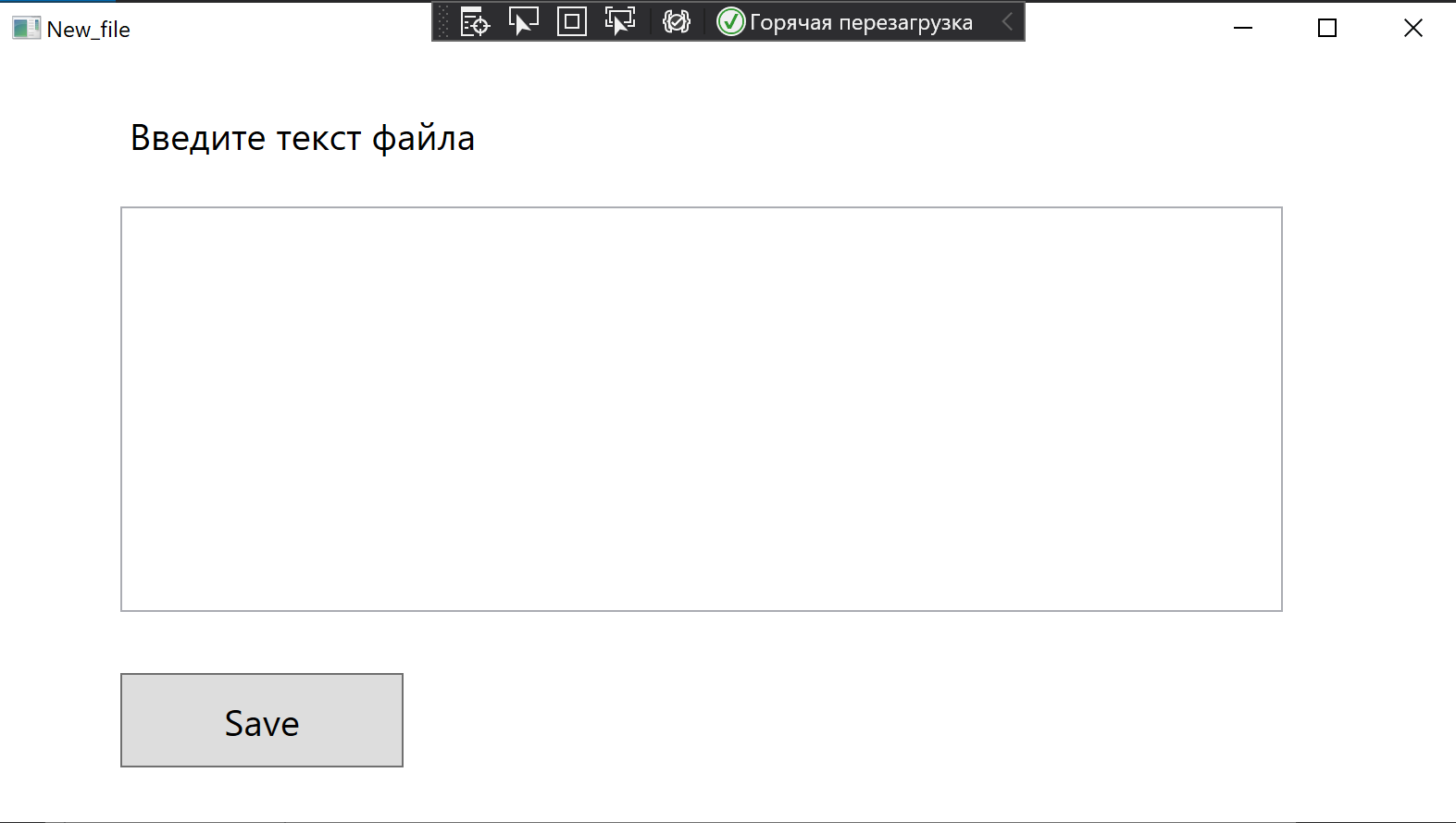


Поиск работает для разных регистров и основан на алгоритме Левенштейна. 

Через текстовое коно вводится расстояние, а через другое слово или набор символов, далее в LIstBox выводится все найденные слова, соответствующие данному расстоянию.



Окно добавления файла.



23.12.2020  
Шкарин Е.А