Лабораторная работа №2

студента группы ИТ – 32

Курбатовой Софьи Андреевны

Выполнение:_____ Защита _____

РЕАЛИЗАЦИЯ АЛГОРИТМОВ СТЕММИНГА.

Цель работы: научиться реализовывать на выбранном языке программирования алгоритмы поиска по тексту однокоренных слов на основе поиска Стеммера Портера.

Содержание работы

Написать программу на выбранном языке программирования, реализующую указанный выше алгоритм для поиска по тексту однокоренных слов Программа должна запрашивать имя входного файла. Оценить трудоемкость рассматриваемых алгоритмов. Результатом работы должен быть файл, содержащий список однокоренных слов.

Ход работы

1. Идея состоит в следующем: не использовать базы основ слов, а последовательно применять ряд правил отсечения окончаний и суффиксов.

Табл 1. Классы окончаний слов

Название класса	Окончания слов
PERFECTIVE	Группа 1*: в, вши, вшись.
GERUND	Группа 2: ив, ивши, ившись, ыв, ывши, ывшись.
ADJECTIVE	ее, ие, ые, ое, ими, ыми, ей, ий, ый, ой, ем, им, ым, ом, его, ого, ему, ому, их, ых, ую, юю, ая, яя, ою, ею.
PARTICIPLE	Группа 1*: ем, нн, вш, ющ, щ.
	Группа 2: ивш, ывш, ующ.
REFLEXIVE	ся, сь.
VERB	Группа 1*: ла, на, ете, йте, ли, й, л, ем, н, ло, но, ет, ют, ны, ть, ешь, нно.
	Группа 2: ила, ыла, ена, ейте, уйте, ите, или, ыли, ей, уй, ил, ыл, им, ым, ен, ило, ыло, ено, ят, ует, уют, ит, ыт, ены, ить, ыть, ишь, ую, ю.
NOUN	а, ев, ов, ие, ье, е, иями, ями, ами, еи, ии, и,ией, ей, ой, ий, й, иям, ям, ием, ем, ам, ом, о, у, ах, иях, ях, ы, ь, ию, ью, ю, ия, ья, я.
SUPERLATIVE	ейш, ейше.
DERIVATIONAL	ост, ость.
ADJECTIVAL	ADJECTIVAL определяется как ADJECTIVE или PARTICIPLE + ADJECTIVE. *Например: бегавшая = бега + вш + ая.*

Алгоритм следует двум правилам:

1. При поиске окончания из всех возможных выбрать наиболее длинное.
2. Все проверки производятся над областью RV (после первой гласной).

Листинг 1.1. Программ реализующая алгоритм поиска.

```
using System.IO;
using System.Text;
using System;
using Stemmer.RU;
using System.Text.RegularExpressions;

namespace Stemmer.RU
{
    class Porter
    {
/*Классы окончаний слов в виде шаблонов*/
        private const string VOWEL = "аеиоуыэюя";
        private const string PERFECTIVEGROUND =
"((ив|ивши|ившись|ыв|ывши|ывшись)|((?<=[ая])(в|вши|вшись)))$";
        private const string REFLEXIVE = "(с[яь])$";
        private const string ADJECTIVE =
"(ее|ие|ые|ое|ими|ыми|ей|ий|ый|ой|ем|им|ым|ом|его|ого|еых|ую|юю|ая|яя|ою|ею)$";
        private const string PARTICIPLE = "((ивш|ывш|ующ)|((?<=[ая])(ем|нн|вш|ющ|щ)))$";
        private const string VERB =
"((ила|ыла|ена|ейте|уйте|ите|или|ыли|ей|уй|ил|ыл|им|ым|ены|ить|ыть|ишь|ую|ю)|((?<=[ая])(ла|на|ете
|йте|ли|й|л|ем|н|ло|но|ет|ют|ны|ть|ешь|нно)))$";
        private const string NOUN =
"(а|ев|ов|ие|ье|е|иями|ями|ами|еи|ии|и|ией|ей|ой|ий|й|и|ы|ь|ию|ью|ю|ия|ья|я)$";
        private const string RVRE = "^(.*?[аеиоуыэюя])(.*)$";
        private const string DERIVATIONAL = "(ост|ость)?";
        private const string SUPERLATIVE = "(ейше|ейш)?";
/*функция выполняющая отсечение окончаний у слова сокращая его до основы*/
        public string Stemm(string word)
        {
            word = word.ToLower();
            word = word.Replace("ё", "е");
            if (IsMatch(word, RVRE))
            {
/*Найти окончание PERFECTIVE GERUND*/
                if (!Replace(ref word, PERFECTIVEGROUND, ""))
                {
                    Replace(ref word, REFLEXIVE, "");
                    if (Replace(ref word, ADJECTIVE, ""))
                    {
                        Replace(ref word, PARTICIPLE, "");
                    }
                    else
                    {
                        if (!Replace(ref word, VERB, ""))
                        {
                            Replace(ref word, NOUN, "");
                        }

                    }
                }
/*отсечение и, если слово заканчивается этой буквой*/
                Replace(ref word, "и$", "");

                if (IsMatch(word, DERIVATIONAL))
                {
                    Replace(ref word, DERIVATIONAL, "");
                }

                if (!Replace(ref word, "ь$", ""))
                {
                    Replace(ref word, SUPERLATIVE, "");
                    Replace(ref word, "нн$", "н");//слово закончилось на нн значит убрать
                }

            }
```

```
            return word;//основа слова
        }

        private bool IsMatch(string word, string matchingPattern)
        {
            return new Regex(matchingPattern).IsMatch(word);
        }

        private bool Replace(ref string replace, string cleaningPattern, string by)
        {
            string original = replace;
/**/
            replace = new Regex(cleaningPattern,
                        RegexOptions.ExplicitCapture |
                        RegexOptions.Singleline
                        ).Replace(replace, by);
            return original != replace;
        }
    }
}

namespace Stemmer
{
    class Program
    {
        static void Main(string[] args)
        {
            string a = "";   string word = "";
            Porter p = new Porter();
            Console.WriteLine("Введите адрес исходного файла: ");
            string path = Console.ReadLine();
            while (true)
            {
                Console.WriteLine("Введите слово:");
                string input = Console.ReadLine();
                if (input != "выйти")
                {  Console.WriteLine("Основа слова " + "input " + ": " + p.Stemm(input));  }
                else  {  break;  }
/*переменная word – это основа слова, которую будем искать в тексте*/
                word = p.Stemm(input);
                FileStream stream = new FileStream(@"C:\test\output.txt", FileMode.Create);
                StreamWriter writer = new StreamWriter(stream);
                try
                {
                    Console.WriteLine();
                    Console.WriteLine("Однокоренные слова в файле:");
                    using (StreamReader sr = new StreamReader(path, Encoding.Default))
                    {
                        string line;
                        while ((line = sr.ReadLine()) != null)
                        {
                            a = line;
                            if (a.IndexOf(word) > -1)
                            {
                                Console.WriteLine(a);
                                writer.WriteLine(a);
                            }
                        }
                    }
                    Console.WriteLine();
                }
                catch (Exception e)
                {  Console.WriteLine(e.Message);  }
                writer.Close();
            }
        }
    }
```

```
}
```

2. Тестирование:

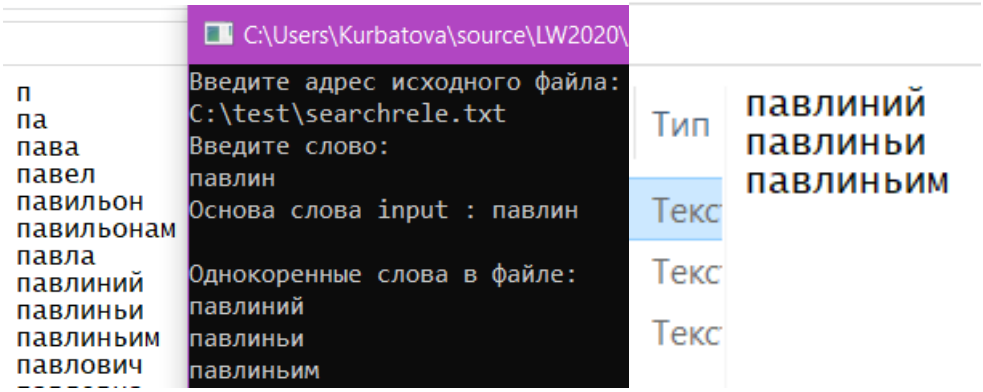

Рис. 2.1. Исходный файл и результат работы программы

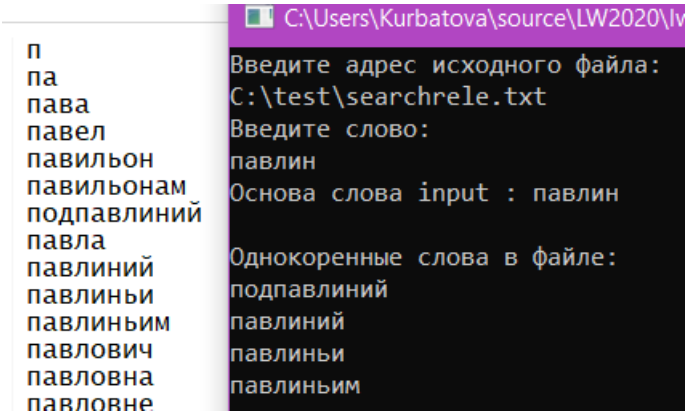

Рис. 2.2. Новая версия словаря и результат работы

Вывод: Таким образом, в ходе выполнения лабораторной работы на языке C# были реализованы алгоритмы поиска по тексту однокоренных слов по шаблону задаваемому пользователем, в текстовом файле, путь к которому также задается пользователем. Результатом выполнения поиска является текстовый файл, содержащий найденные значения.