Санкт-Петербургский Национальный Исследовательский Университет
Информационных Технологий, Механики и Оптики

Лабораторная работа №9

по дисциплине

«Алгоритмы и структуры данных»

Выполнил: Студент группы P3217

Сергачев Данила Дмитриевич

Преподаватели:

Романов Алексей Андреевич и

Волчек Дмитрий Геннадьевич

Санкт-Петербург

2019 г

Задание №1

Наивный поиск подстроки в строке

1.0 из 1.0 балла (оценивается)

Имя входного файла:	input.txt
Имя выходного файла:	output.txt
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Даны строки p и t. Требуется найти все вхождения строки p в строку t в качестве подстроки.

Решение

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

namespace Week9
{
    class Task1
    {
        public static void Main(string[] args)
        {
            using (StreamWriter sw = new StreamWriter("output.txt"))
            {
                string[] stdin = File.ReadAllLines("input.txt");
                foreach (var line in stdin)
                {
                    var text = Regex.Replace(line, @"\s", string.Empty);
                    if (text.Length < 3)
                    {
                        sw.WriteLine("0");
                        continue;
                    }

                    var right = text.GroupBy(c => c).ToDictionary(g => g.Key, g => new
S(0, g.Count()));
                    var left = new Dictionary<char, S>();

                    var cc = text[0];
                    var entry = right[cc];
                    left[cc] = entry;
                    entry.left++;
                    entry.right--;

                    var count = text.Length - 1;
                    ulong s = 0;
                    for (int i = 1; i < count; i++)
                    {
                        cc = text[i];
                        entry = right[cc];
                        entry.right--;
```

```
                    foreach (var kv in left)
                        s += kv.Value.Muls;

                    entry.left++;
                    left[cc] = entry;
                }

                sw.WriteLine(s);
            }
        }
    }

    class S
    {
        public int left, right;

        public ulong Muls { get => (ulong)this.left * (ulong)this.right; }

        public S(int left, int right)
        {
            this.left = left;
            this.right = right;
        }
    }
}
}
```

Результат работы

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.078	11542528	20003	48890
1	OK	0.015	10416128	14	7
2	OK	0.015	10354688	6	5
3	OK	0.031	10424320	6	3
4	OK	0.031	10375168	7	7
5	OK	0.031	10362880	7	3
6	OK	0.031	10391552	9	7
7	OK	0.031	10379264	10	5

Задание №2

Карта

1.0 из 1.0 балла (оценивается)

Имя входного файла:	input.txt
Имя выходного файла:	output.txt
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

> Даже самый последний матрос знает, что мы едем искать сокровища. Не нравится мне всё это!
>
> Капитан Смоллетт

Команда Александра Смоллетта догадалась, что сокровища находятся на x шагов восточнее красного креста, однако определить значение числа она не смогла. По возвращению на материк Александр Смоллетт решил обратиться за помощью в расшифровке послания к знакомому мудрецу. Мудрец поведал, что данное послание таит за собой некоторое число. Для вычисления этого числа необходимо было удалить все пробелы между словами, а потом посчитать количество способов вычеркнуть все буквы кроме трех так, чтобы полученное слово из трех букв одинаково читалось слева направо и справа налево.

Александр Смоллетт догадывался, что число, зашифрованное в послании, и есть число x. Однако, вычислить это число у него не получилось.

После смерти капитана карта была безнадежно утеряна до тех пор, пока не оказалась в ваших руках. Вы уже знаете все секреты, осталось только вычислить число x.

Решение

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;
using System.Text.RegularExpressions;

namespace Week9
{
    class Task2
    {
        static void Main(string[] args)
        {
            using (StreamWriter sw = new StreamWriter("output.txt"))
            {
                string text =  File.ReadAllText("input.txt").Replace(" ", string.Empty);

                if(text.Length < 3)
                {
                    sw.WriteLine("0");
                }
                else
                {
                    var rightDictionary = text
                        .GroupBy(s => s)
                        .ToDictionary(s => s.Key, s => new Symbol(0, s.Count()));

                    var leftDictionary = new Dictionary<char, Symbol>();

                    long totalCount = 0;
```

```
                    for(int i = 0; i < text.Length; i++)
                    {
                        char curSymbol = text[i];
                        Symbol symbol = rightDictionary[curSymbol];
                        symbol.rightEntry--;

                        foreach(var entry in leftDictionary)
                        {
                            totalCount += entry.Value.NumbersPalindromes;
                        }
                        symbol.leftEntry++;
                        leftDictionary[curSymbol] = symbol;
                    }

                    sw.WriteLine(totalCount);
                }
            }
        }

        private class Symbol
        {
            public long leftEntry, rightEntry;

            public Symbol(long leftEntry, long rightEntry)
            {
                this.leftEntry = leftEntry;
                this.rightEntry = rightEntry;
            }

            public long NumbersPalindromes { get => this.leftEntry * this.rightEntry;  }
        }
    }
}
```

Результат работы

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.171	15233024	300002	18
1	OK	0.046	12316672	10	3
2	OK	0.046	12357632	34	5
3	OK	0.046	12337152	5	3
4	OK	0.046	12283904	6	3
5	OK	0.046	12296192	7	3
6	OK	0.046	12345344	9	4
7	OK	0.046	12357632	7	3