ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

Факультет программной инженерии и компьютерной техники

Лабораторная работа №9

Девятая неделя

Выполнил:

Жумиков Егор Олегович

Преподаватели:

Романов Алексей Андреевич

Волчек Дмитрий Геннадьевич

Санкт-Петербург, 2019

Оглавление

Задача «Наивный поиск подстроки в строке»

Условие

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

Формат входного файла

Выведите последовательно результат выполнения всех операций «?». Следуйте формату выходного файла из примера.

Формат выходного файла

В первой строке выведите число вхождений строки p в строку t. Во второй строке выведите в возрастающем порядке номера символов строки t, с которых начинаются вхождения p. Символы нумеруются с единицы.

Решение

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

namespace ItmoAlgos
{
    public class Program
    {
        private static string[] _input;
        private static int _currentLineIndex;
        private const long E15 = 1000000000000000;

        private static string ReadLine()
        {
            return _input[_currentLineIndex++];
        }

        public static int Hash(string s, int m)
        {
            int rv = 0;
            foreach (char c in s)
            {
                rv = m * rv + c;
            }
            return rv;
        }

        public static void Main(string[] args)
        {
            _input = File.ReadAllLines("input.txt");

            string
                p = ReadLine(),
                t = ReadLine();

            var occurences = new List<int>();

            for (int i = 0; i <= t.Length - p.Length; i++)
            {
                if (t.Substring(i).StartsWith(p))
                {
                    occurences.Add(i + 1);
```

```
                }
            }


            File.WriteAllText("output.txt", $"{occurences.Count}\n{string.Join(" ",
occurences.Select(o => o.ToString()))}");
        }
    }
}
```

Результат

Верное решение!
Результаты работы Вашего решения

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.187	13426688	20003	48888
1	OK	0.031	11517952	14	5
2	OK	0.046	11612160	6	3

Задача «Карта»

Условие

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

Капитан Смоллетт

В далеком 1744 году во время долгого плавания в руки капитана Александра Смоллетта попала древняя карта с указанием местонахождения сокровищ. Однако расшифровать ее содержание было не так уж и просто.

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.

Формат входного файла

В первой строке входного файла находится строго положительное целое число операций N, не превышающее $5 \cdot 10^5$. В каждой из последующих N строк находится одна из следующих операций:

- `get` x — если ключ x есть в множестве, выведите соответствующее ему значение, если нет, то выведите `<none>`.
- `prev` x — вывести значение, соответствующее ключу, находящемуся в ассоциативном массиве, который был вставлен позже всех, но до x, или `<none>`, если такого нет или в массиве нет x.
- `next` x — вывести значение, соответствующее ключу, находящемуся в ассоциативном массиве, который был вставлен раньше всех, но после x, или `<none>`, если такого нет или в массиве нет x.
- `put` x y — поставить в соответствие ключу x значение y. При этом следует учесть, что:
  - если, независимо от предыстории, этого ключа на момент вставки в массиве не было, то он считается только что вставленным и оказывается самым последним среди добавленных элементов — то есть, вызов `next` с этим же ключом сразу после выполнения текущей операции `put` должен вернуть `<none>`;
  - если этот ключ уже есть в массиве, то значение необходимо изменить, и в этом случае ключ не считается вставленным еще раз, то есть, не меняет своего положения в порядке добавленных элементов.
- `delete` x — удалить ключ x. Если ключа x в ассоциативном массиве нет, то ничего делать не надо.

Ключи и значения — строки из латинских букв длиной не менее одного и не более 20 символов.

Формат выходного файла

Выведите одно число X — число способов вычеркнуть из послания все буквы кроме трех так, чтобы оставшееся слово одинаково читалось слева направо и справа налево.

Решение

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

namespace ItmoAlgos
{
    public class Program
    {
        private static string[] _input;
        private static int _currentLineIndex;
        private const long E15 = 1000000000000000;

        private static string ReadLine()
        {
            return _input[_currentLineIndex++];
        }

        public static void Main(string[] args)
        {
            _input = File.ReadAllLines("input.txt");

            string map = ReadLine().Replace(" ", "");
            long count = 0;

            var onTheWay = new long[map.Length + 1][];
            for (int i = 0; i ⩽ map.Length; i++)
            {
                onTheWay[i] = new long[26];
            }

            for (int i = 1; i ⩽ map.Length; i++)
            {
                Array.Copy(onTheWay[i - 1], onTheWay[i], 26);
                onTheWay[i][map[i - 1] - 'a']++;
            }

            for (int i = 1; i < map.Length - 1; i++)
            {
                for (int j = 0; j < 26; j++)
                {
                    count += (onTheWay[map.Length][j] - onTheWay[i +
1][j]) * onTheWay[i][j];
                }
            }

            File.WriteAllText("output.txt", count.ToString());
        }
    }
```

}

Результат

Верное решение!
Результаты работы Вашего решения

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.203	84246528	300002	16
1	OK	0.015	10076160	10	1