Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики

Кафедра информатики и прикладной математики

Лабораторная работа №9
Дисциплина «Алгоритмы и структуры данных»

Выполнил:
Молодецкий Арсений Алексеевич
Группа P3217

Санкт-Петербург
2019

Задание №1

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

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

namespace tmp
{
    public class Program
    {
        private static StreamWriter _out;

        private static void Main(string[] args)
        {

            _out = new StreamWriter("output.txt");
            Console.SetOut(_out);
            var input = File.ReadAllLines("input.txt");
            string t = input[1];
            string p = input[0];
            int i = 0;
            int i2 = 0;
            var indexes = new List<int>();

            while( i < t.Length) {
                int j = 0;
                for (; j < p.Length && i+j < t.Length && t[i + j] == p[j];){
                    j++;
                }
                if (j == p.Length)
                {
                    indexes.Add(i+1);
                }
                i++;
            }
            Console.WriteLine(indexes.Count);
            Console.WriteLine(String.Join(" ", indexes));
            DisposeIO();
        }

        private static void DisposeIO()
        {
            _out?.Dispose();
        }
    }
}
```

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.140	11436032	20003	48891
1	OK	0.031	10493952	14	8
2	OK	0.015	10518528	6	6
3	OK	0.031	10498048	6	5
4	OK	0.015	10489856	7	8
5	OK	0.015	10502144	7	5
6	OK	0.031	10539008	9	8
7	OK	0.015	10551296	10	6
8	OK	0.015	10489856	3004	5
9	OK	0.031	10534912	3028	8
10	OK	0.015	10498048	2656	430
11	OK	0.015	10653696	2005	8896
12	OK	0.031	10489856	4003	8
13	OK	0.031	10547200	3004	5
14	OK	0.031	10514432	2252	1851
15	OK	0.015	10498048	2021	187
16	OK	0.031	10813440	2008	8885
17	OK	0.031	10563584	3004	3905
18	OK	0.031	10555392	2670	5
19	OK	0.031	10547200	3028	8
20	OK	0.015	10502144	2404	692
21	OK	0.015	10805248	2005	8900
22	OK	0.031	10493952	4003	8
23	OK	0.031	10641408	2670	5
24	OK	0.015	10506240	2252	1887
25	OK	0.031	10489856	2022	191
26	OK	0.046	10645504	2008	8885
27	OK	0.031	10559488	3004	3905
28	OK	0.031	10522624	5337	5
29	OK	0.031	10514432	5028	9
30	OK	0.031	10518528	4372	649
31	OK	0.031	10883072	4005	18900
32	OK	0.046	10510336	8003	8
33	OK	0.046	10518528	5337	5
34	OK	0.015	10616832	4804	3481
35	OK	0.031	10551296	4015	790
36	OK	0.015	10825728	4008	18865
37	OK	0.046	10739712	6004	8905
38	OK	0.031	10514432	5337	5
39	OK	0.031	10502144	5028	9
40	OK	0.015	10530816	4477	787
41	OK	0.031	10825728	4005	18895
42	OK	0.031	10526720	8003	8
43	OK	0.031	10555392	5337	5
44	OK	0.031	10612736	4572	3975
45	OK	0.031	10506240	4015	398
46	OK	0.031	10878976	4008	18885
47	OK	0.031	10674176	6004	8905
48	OK	0.046	10518528	9004	5
49	OK	0.015	10575872	7028	14
50	OK	0.031	10567680	7179	661
51	OK	0.031	11005952	6005	28900
52	OK	0.062	10539008	12003	8
53	OK	0.062	10592256	8004	5
54	OK	0.031	10674176	6752	5679
55	OK	0.015	10543104	6015	1205
56	OK	0.031	11001856	6008	28885
57	OK	0.062	10764288	9004	13905
58	OK	0.031	10608640	9004	5
59	OK	0.031	10518528	7028	9
60	OK	0.031	10584064	6470	507
61	OK	0.031	11005952	6005	28900
62	OK	0.046	10543104	12003	8
63	OK	0.046	10563584	8004	5
64	OK	0.031	10653696	8004	4481
65	OK	0.031	10534912	6016	609
66	OK	0.031	11038720	6008	28885
67	OK	0.062	10760192	9004	13905
68	OK	0.062	10592256	12004	5
69	OK	0.031	10522624	9028	14
70	OK	0.031	10608640	9920	440
71	OK	0.031	11132928	8005	38900
72	OK	0.109	10620928	16003	8
73	OK	0.093	10571776	12004	5
74	OK	0.031	10670080	8728	8377
75	OK	0.031	10563584	8017	1624
76	OK	0.031	11141120	8008	38845
77	OK	0.078	10846208	12004	18905
78	OK	0.062	10539008	12004	5
79	OK	0.015	10539008	9028	18
80	OK	0.015	10625024	10660	351
81	OK	0.015	11145216	8005	38900
82	OK	0.093	10551296	16003	8
83	OK	0.062	10539008	10670	5
84	OK	0.031	10625024	10004	6770
85	OK	0.046	10547200	8022	813
86	OK	0.031	11243520	8008	38885
87	OK	0.093	10862592	12004	18905
88	OK	0.078	10612736	15004	5
89	OK	0.031	10551296	11028	18
90	OK	0.031	10555392	10925	666
91	OK	0.015	11403264	10005	48886
92	OK	0.109	10649600	20003	8
93	OK	0.078	10551296	13337	5
94	OK	0.031	10682368	12504	8257
95	OK	0.031	10567680	10020	1023
96	OK	0.046	11370496	10008	48885
97	OK	0.109	10973184	15004	23905
98	OK	0.078	10625024	15004	5
99	OK	0.031	10551296	11028	18
100	OK	0.031	10596352	11004	499
101	OK	0.031	11382784	10005	48891
102	OK	0.140	10731520	20003	8
103	OK	0.078	10563584	13337	5
104	OK	0.046	10788864	10912	10927
105	OK	0.031	10571776	10015	2043
106	OK	0.031	11436032	10008	48885
107	OK	0.109	10977280	15004	23905

Задание №2

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.

```
using System;
using System.IO;

namespace tmp
{
    public class Program
    {
        private static StreamWriter _out;

        private static void Main(string[] args)
        {
            var input = File.ReadAllLines("input.txt")[0].Replace(" ","");
            int[] countLetters = new int[26];
            long[] difference = new long[26];
            int[] lastIndexes = new int[26];

            long count = 0;
            for( int i = 0; i < input.Length; i++)
            {
                int index = input[i] - 'a';
                int length = i - lastIndexes[index] - 1;
                if ( countLetters[index] > 1)
                {
                    difference[index] += countLetters[index] - 1;
                }
                difference[index] += length * (countLetters[index]);
                countLetters[index]++;

                count += difference[index];


                lastIndexes[index] = i;
            }

            _out = new StreamWriter("output.txt");
            Console.SetOut(_out);
            Console.WriteLine(count);
            DisposeIO();
```

```
        }

        private static void DisposeIO()
        {
            _out?.Dispose();
        }
    }
}
```

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.062	11964416	300002	18
1	OK	0.015	10063872	10	3
2	OK	0.046	10063872	34	5
3	OK	0.015	10055680	5	3
4	OK	0.031	10043392	6	3
5	OK	0.031	10059776	7	3
6	OK	0.031	10018816	9	4
7	OK	0.015	10076160	7	3
8	OK	0.015	10035200	7	3
9	OK	0.015	10039296	13	4
10	OK	0.046	10043392	202	8
11	OK	0.031	10027008	202	8
12	OK	0.031	10018816	202	8
13	OK	0.015	10063872	202	8
14	OK	0.015	10047488	202	7
15	OK	0.031	10059776	202	7
16	OK	0.031	10059776	202	7
17	OK	0.015	10043392	202	9
18	OK	0.015	10084352	5002	13
19	OK	0.031	10067968	5002	13
20	OK	0.031	10072064	5002	13
21	OK	0.015	10051584	5002	13
22	OK	0.015	10104832	5002	13
23	OK	0.015	10080256	5002	13
24	OK	0.031	10158080	5002	13
25	OK	0.062	10067968	5002	13
26	OK	0.031	10080256	5002	13
27	OK	0.031	10067968	5002	13
28	OK	0.031	10113024	5002	11
29	OK	0.015	10084352	5002	11
30	OK	0.031	10080256	5002	11
31	OK	0.031	10153984	5002	11
32	OK	0.031	10117120	5002	11
33	OK	0.031	11264000	300002	18
34	OK	0.015	11259904	300002	18
35	OK	0.046	11296768	300002	18
36	OK	0.031	11292672	300002	18
37	OK	0.031	11304960	300002	18
38	OK	0.015	11280384	300002	18
39	OK	0.031	11255808	300002	17
40	OK	0.031	11304960	300002	17
41	OK	0.031	11337728	300002	17
42	OK	0.031	11329536	300002	17
43	OK	0.031	11960320	300002	17
44	OK	0.031	11960320	300002	17
45	OK	0.031	11964416	300002	17
46	OK	0.031	11866112	300002	17
47	OK	0.062	11907072	300002	17
48	OK	0.015	11280384	300002	17
49	OK	0.015	11268096	300002	17
50	OK	0.015	11272192	300002	17
51	OK	0.015	11251712	300002	17
52	OK	0.031	11304960	300002	17