Министерство образования и науки Российской Федерации

САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04 Программная инженерия

Дисциплина «Алгоритмы и структуры данных»

ОТЧЁТ

по лабораторной работе №9 (Week 9 Openedu)

Студент Дунаев Алексей Игоревич

Группа P3217

Преподаватель Муромцев Дмитрий Ильич

Санкт-Петербург

2019 г.

Содержание

Задача 1 Наивный поиск подстроки в строке

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.

Формат входного файла

Первая строка входного файла содержит p, вторая — t ($1 \le p, t \le 10^4$). Строки состоят из букв латинского алфавита.

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.

Примеры

input.txt	output.txt
aba abaCaba	2 1 5

Исходный код к задаче 1

```
import re
fo= open("output.txt","w")
fi = open('input.txt', 'r')
sub = fi.readline().strip()
string = fi.readline().strip()
ans = [m.start(0) + 1 for m in re.finditer('(?='+sub+')', string)]
fo.write(str(len(ans)) + '\n')
fo.write(' '.join(str(r) for r in ans))
```

Бенчмарк к задаче 1

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.406	11173888	20003	48889
1	OK	0.046	10072064	14	6
2	OK	0.031	10031104	6	4
3	OK	0.046	9928704	6	3

4	OK	0.046	10006528	7	6
5	OK	0.046	9949184	7	3
6	OK	0.046	10022912	9	6
7	OK	0.046	10006528	10	4
8	OK	0.046	9973760	3004	3
9	OK	0.046	10096640	3028	6
10	OK	0.046	10018816	2656	428
11	OK	0.078	10256384	2005	8894
12	OK	0.046	10223616	4003	6
13	OK	0.062	10096640	3004	3
14	OK	0.046	9953280	2252	1849
15	OK	0.046	10018816	2021	185
16	OK	0.046	10125312	2008	8883
17	OK	0.062	10080256	3004	3903
18	OK	0.046	10018816	2670	3
19	OK	0.046	10108928	3028	6
20	OK	0.046	9986048	2404	690
21	OK	0.078	10121216	2005	8898
22	OK	0.046	10166272	4003	6
23	OK	0.062	10051584	2670	3
24	OK	0.046	9990144	2252	1885
25	OK	0.046	10027008	2022	189
26	OK	0.046	10063872	2008	8883
27	OK	0.062	10092544	3004	3903
28	OK	0.062	10137600	5337	3
29	OK	0.062	10063872	5028	7
30	OK	0.046	9969664	4372	647
31	OK	0.046	10330112	4005	18898
32	OK	0.093	10387456	8003	6
33	OK	0.078	10170368	5337	3

34	OK	0.093	10080256	4804	3479
35	OK	0.031	10039296	4015	788
36	OK	0.046	10448896	4008	18863
37	OK	0.109	10375168	6004	8903
38	OK	0.062	10186752	5337	3
39	OK	0.046	10047488	5028	7
40	OK	0.046	9998336	4477	785
41	OK	0.062	10412032	4005	18893
42	OK	0.093	10416128	8003	6
43	OK	0.078	10153984	5337	3
44	OK	0.078	10059776	4572	3973
45	OK	0.046	9994240	4015	396
46	OK	0.078	10432512	4008	18883
47	OK	0.125	10428416	6004	8903
48	OK	0.109	10244096	9004	3
49	OK	0.078	10104832	7028	12
50	OK	0.046	10129408	7179	659
51	OK	0.062	10698752	6005	28898
52	OK	0.156	10547200	12003	6
53	OK	0.078	10280960	8004	3
54	OK	0.062	10113024	6752	5677
55	OK	0.062	9961472	6015	1203
56	OK	0.062	10657792	6008	28883
57	OK	0.171	10457088	9004	13903
58	OK	0.093	10268672	9004	3
59	OK	0.046	10092544	7028	7
60	OK	0.046	10018816	6470	505
61	OK	0.062	10612736	6005	28898
62	OK	0.171	10702848	12003	6

63	OK	0.109	10231808	8004	3
64	OK	0.109	10326016	8004	4479
65	OK	0.046	9981952	6016	607
66	OK	0.046	10653696	6008	28883
67	OK	0.218	10391552	9004	13903
68	OK	0.140	10399744	12004	3
69	OK	0.046	10129408	9028	12
70	OK	0.062	10211328	9920	438
71	OK	0.046	10838016	8005	38898
72	OK	0.250	10850304	16003	6
73	OK	0.187	10334208	12004	3
74	OK	0.062	10276864	8728	8375
75	OK	0.046	10092544	8017	1622
76	OK	0.046	10878976	8008	38843
77	OK	0.265	10539008	12004	18903
78	OK	0.140	10440704	12004	3
79	OK	0.046	10117120	9028	16
80	OK	0.046	10244096	10660	349
81	OK	0.046	10838016	8005	38898
82	OK	0.281	10833920	16003	6
83	OK	0.140	10203136	10670	3
84	OK	0.078	10399744	10004	6768
85	OK	0.046	10039296	8022	811
86	OK	0.062	10776576	8008	38883
87	OK	0.281	10547200	12004	18903
88	OK	0.187	10530816	15004	3
89	OK	0.046	10129408	11028	16
90	OK	0.046	10084352	10925	664
91	OK	0.062	11153408	10005	48884

92	OK	0.343	11096064	20003	6
93	OK	0.203	10309632	13337	3
94	OK	0.093	10448896	12504	8255
95	OK	0.046	10006528	10020	1021
96	OK	0.062	11079680	10008	48883
97	OK	0.390	10649600	15004	23903
98	OK	0.187	10661888	15004	3
99	OK	0.046	10076160	11028	16
100	OK	0.062	10100736	11004	497
101	OK	0.062	11108352	10005	48889
102	OK	0.328	11173888	20003	6
103	OK	0.187	10342400	13337	3
104	OK	0.078	10350592	10912	10925
105	OK	0.046	9969664	10015	2041
106	OK	0.062	11153408	10008	48883
107	OK	0.406	10702848	15004	23903

Задача 2. Карта

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

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 единственной строке входного файла дано послание, написанное на карте. Длина послания не превышает $3 * 10^5$. Гарантируется, что послание может содержать только строчные буквы английского алфавита и пробелы. Также гарантируется, что послание не пусто. Послание не может начинаться с пробела или заканчиваться им.

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.

Примеры

input.txt	output.txt
treasure	8
you will never find the treasure	146

Исходный код к задаче 2

```
#include <fstream>
#include <vector>
#include <iostream>
#include <algorithm>
#include <string>
#include <unordered_map>
#include <list>
#include <map>
using namespace std;
/**
#include "edx-io.hpp"
#define cout io
#define cin io
/**/
long long calc_distance(vector<long> pos) {
    long long sum = 0;
    long long temp;
    long k = pos.size() - 1;
    for (long i = pos.size() - 1; i >= 0; i--) {
        temp = (long long)k * pos[i];
```

```
            sum += temp - i;
            k -= 2;
        }
        return sum;
}
int main() {
        ifstream fi("input.txt");
        ofstream fo("output.txt");
        string st = "";
        string tmp;
        while (!fi.eof()) {
            fi >> tmp;
            if (tmp != "") {
                st += tmp;
            }
            tmp = "";
        }
        long long ans = 0;
        map<char, vector<long>> lett;
        for (long i = 0; i < st.length(); i++) {
            lett[st[i]].push_back(i);
        }
        for (char i = 'a'; i <= 'z'; i++) {
            if (lett.count(i) && lett[i].size() > 1) {
                ans += calc_distance(lett[i]);
            }
        }
        fo << ans;
        fo << "\n";
        return 0;
}
```

Бенчмарк к задаче 2

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.031	5976064	300002	16
1	OK	0.000	2363392	10	1
2	OK	0.015	2371584	34	3
3	OK	0.015	2355200	5	1
4	OK	0.000	2359296	6	1
5	OK	0.000	2371584	7	1
6	OK	0.000	2359296	9	2
7	OK	0.000	2379776	7	1
8	OK	0.000	2375680	7	1
9	OK	0.000	2355200	13	2
10	OK	0.015	2359296	202	6

11	OK	0.015	2367488	202	6
12	OK	0.000	2359296	202	6
13	OK	0.015	2355200	202	6
14	OK	0.015	2367488	202	5
15	OK	0.000	2379776	202	5
16	OK	0.015	2367488	202	5
17	OK	0.015	2351104	202	7
18	OK	0.000	2347008	5002	11
19	OK	0.000	2363392	5002	11
20	OK	0.015	2359296	5002	11
21	OK	0.015	2347008	5002	11
22	OK	0.000	2367488	5002	11
23	OK	0.000	2347008	5002	11
24	OK	0.015	2347008	5002	11
25	OK	0.000	2347008	5002	11
26	OK	0.000	2342912	5002	11
27	OK	0.000	2367488	5002	11
28	OK	0.000	2453504	5002	9
29	OK	0.015	2465792	5002	9
30	OK	0.015	2453504	5002	9
31	OK	0.000	2453504	5002	9
32	OK	0.000	2449408	5002	9
33	OK	0.015	4952064	300002	16
34	OK	0.000	5976064	300002	16
35	OK	0.015	5971968	300002	16
36	OK	0.015	5971968	300002	16
37	OK	0.015	5971968	300002	16
38	OK	0.015	5971968	300002	16
39	OK	0.015	4808704	300002	15
40	OK	0.015	4870144	300002	15

41	OK	0.031	4694016	300002	15
42	OK	0.031	4763648	300002	15
43	OK	0.015	4423680	300002	15
44	OK	0.015	4395008	300002	15
45	OK	0.015	4435968	300002	15
46	OK	0.031	4415488	300002	15
47	OK	0.031	4460544	300002	15
48	OK	0.031	4657152	300002	15
49	OK	0.031	4739072	300002	15
50	OK	0.015	4763648	300002	15
51	OK	0.015	4796416	300002	15
52	OK	0.015	4702208	300002	15