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

ПО ДИСЦИПЛИНЕ «Алгоритмы и структуры данных»

Выполнил: Мещеряков Владимир Алевтинович

Группа: P3218

Преподаватель: Романов Алексей Андреевич

Волчек Дмитрий Геннадьевич

Санкт-Петербург

2019 г.

Week 9

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.

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

```
#include <string>
#include "edx-io.hpp"

using namespace std;


int main() {
    string p;
    string t;

    int count = 0;

    io >> p >> t;
    int* arr = new int[t.length()];
    if(t.length() >= p.length()){
    for (int i = 0; i < t.length()-p.length()+1; ++i) {
        if (p[0] == t[i]){
            bool ok = true;
            for(int j = 0; j < p.length(); ++j){
```

```
                if(p[j] != t[i+j]){
                    ok = false;
                    break;
                }
            }
            if(ok){
                arr[count] =  i+1;
                count++;
            }
        }
    }
    }
   io << count << '\n';
    for (int i = 0; i < count; i++)
    {
            io << arr[i] << " ";
        }
    return 0;
}
```

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.046	2281472	20003	48890
1	OK	0.000	2224128	14	7
2	OK	0.000	2224128	6	5
3	OK	0.000	2224128	6	3
4	OK	0.015	2240512	7	7
5	OK	0.000	2224128	7	3

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

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 $3 \cdot 10^5$. 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

```
#include <fstream>
#include <string>
#include <map>
#include <vector>

using namespace std;
//Находит сумму дистанций - разностей позиций каждого с каждым
//Формула: сумм(((n-1) - 2(i-1)) * pos[i - 1])
//i изменяется так, чтобы pos[i] > pos[i+1]
long long calculate_sum_distance(vector<long> positions) {
	long long sum = 0;
	long long temp;
	//Коэффициент слагаемого (n-1) - 2(i-1)
	long k = positions.size() - 1;
	for (long i = positions.size() - 1; i >= 0; i--) {
		temp = (long long)k * positions[i];
		sum += temp;
```

```
			k -= 2;
		}
		//Точки не включены, поэтому необходимо из каждой разности вычесть единицу
		for (long i = 1; i < positions.size(); i++)
		{
			sum -= i;
		}
		return sum;
}


int main() {

		ifstream input("input.txt");
		ofstream output("output.txt");

		string P = "";
		string tmp;
		//Слепливаем строку в одну без пробелов
		while (!input.eof()) {
			input >> tmp;
			if (tmp != "") {
				P += tmp;
			}
			tmp = "";
		}
		//Ассоциативный массив списков, отсортированный по алфавиту
		map<char, vector<long>> letters;
		//Позицию каждой буквы в строке добавляем в соответствующий список
		for (long i = 0; i < P.length(); i++) {
			letters[P[i]].push_back(i);
		}
		long long counter = 0;
		//Для каждого списка с количесвом элементов > 1 находим сумму разностей его элементов
		for (char i = 'a'; i <= 'z'; i++) {
			if (letters.count(i) && letters[i].size() > 1) {
				counter += calculate_sum_distance(letters[i]);
			}
		}

		output << counter;
		return 0;
}
```

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.031	5976064	300002	16
1	OK	0.000	2580480	10	1
2	OK	0.015	2580480	34	3
3	OK	0.000	2572288	5	1
4	OK	0.000	2580480	6	1
5	OK	0.015	2572288	7	1
6	OK	0.000	2576384	9	2