

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего
образования

САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ ИТМО

Факультет систем управления и робототехники

Лабораторная работа № 3

"Задачи 1207, 1322, 1444"

по дисциплине Алгоритмы и структуры данных

Выполнила: студентка гр. **R3238**
поток **2.1**

Нечаева А. А.

Преподаватель: *Тропченко Андрей Александрович*

Санкт-Петербург, 2024

1 Цель

Разработать и реализовать алгоритмы для решения задач 1207, 1322 и 1444.

2 Задача 1207

1207. Медиана на плоскости

Ограничение времени: 0.5 секунды
Ограничение памяти: 64 МБ

На плоскости находятся N точек (N чётно). Никакие три точки не лежат на одной прямой. Ваша задача — выбрать две точки так, что прямая линия, проходящая через них, делит множество точек на две части одинакового размера.

Исходные данные

Первая строка содержит целое число N ($4 \leq N \leq 10\,000$). Каждая из следующих N строк содержит пары целых чисел x_i, y_i ($-10^6 \leq x_i, y_i \leq 10^6$) — координаты i -й точки.

Результат

Выведите номера выбранных точек.

Пример

исходные данные	результат
4 0 0 1 0 0 1 1 1	1 4

Автор задачи: Павел Атнашев
Источник задачи: Соревнование команд УрГУ, март 2002

Рис. 1. Условие задачи 1207.

2.1 Основная идея

Задача сводится к поиску максимальной суммы подпоследовательности последовательности p_i .

2.2 Краткое описание алгоритма

- 1. Входные данные:** первая строка содержит целое число N ($4 \leq N \leq 10000$). Каждая из следующих N строк содержит пары целых чисел x_i, y_i ($-10^6 \leq x_i, y_i \leq 10^6$) — координаты i -й точки.
- 2.** Для разделения плоскости на 2 части необходимо найти точку, которая расположена левее всех остальных.
- 3.** Далее определим угол прямой, проведенной до каждой из оставшихся

точек и отсортируем точки по величине угла.

4. Теперь мы можем определить точку, через которую проходит прямая, разделяющая точки на плоскости пополам.

5. Выходные данные: порядковые номера 2-х точек, прямая через которые разделит все точки плоскости на 2 равные половины.

2.3 Листинг

Листинг 1. Исходный код для 1207

```
1 #include <iostream>
2 #include <cmath>
3
4 #define PI 3.14159265358979323846
5
6 // new struct for points, it keeps x, y, number of the point
   and angle
7 struct point_struct {
8     int x;
9     int y;
10    double angle;
11    int num;
12 };
13
14 point_struct data[10000];
15
16 bool compare(point_struct a, point_struct b) {
17     return a.angle < b.angle;
18 }
19
20 void quick_sort(int left, int right){
21     int i = left;
22     int j = right;
23
24     point_struct median = data[(left + right) / 2];
25
26     while (i <= j) {
27         while (compare(data[i], median)) {
28             ++i;
29         }
30         while (compare(median, data[j])) {
31             --j;
32         }
```

```
33         if (i <= j) {
34             std::swap(data[i], data[j]);
35             ++i;
36             --j;
37         }
38     }
39     if (i < right) {
40         quick_sort(i, right);
41     }
42     if (left < j) {
43         quick_sort(left, j);
44     }
45 };
46
47 int main() {
48     int n;
49     std::cin >> n;
50
51     int first_x = 1000000;
52     int first_id = 0;
53
54     for (int i = 0; i < n; ++i) {
55         int p_x;
56         int p_y;
57
58         std::cin >> p_x >> p_y;
59
60         if (p_x < first_x) {
61             first_id = i;
62             first_x = p_x;
63         }
64
65         data[i].x = p_x;
66         data[i].y = p_y;
67         data[i].num = i;
68     }
69
70     // here we analyze the angles
71     for (int i = 0; i < n; ++i) {
72         if (data[i].num == first_id) {
73             data[i].angle = -360;
74         } else {
75             if (data[i].x == data[first_id].x) {
```

```

76         if (data[i].y > data[first_id].y) {
77             data[i].angle = 90;
78         } else {
79             data[i].angle = -90;
80         }
81     } else {
82         data[i].angle = atan((double) (data[i].y -
            data[first_id].y) / (data[i].x - data[
            first_id].x)) * 180.0 / PI;
83     }
84 }
85 }
86
87 quick_sort(0, n - 1);
88
89 std::cout << first_id + 1 << " " << data[n / 2].num + 1 <<
    std::endl;
90
91 return 0;
92 }

```

2.4 Результат

Автор: [misDragon](#) • Задача: [Медиана на плоскости](#)

ID	Дата	Автор	Задача	Язык	Результат проверки	№ теста	Время работы	Выделено памяти
10587105	16:55:35 15 мар 2024	misDragon	1207. Медиана на плоскости	Clang++ 17 x64	Accepted		0.046	620 KB

Рис. 2. Результат отправки задачи 1207.

3 Задача 1322

1322. Шпион

Ограничение времени: 0.25 секунды
Ограничение памяти: 64 МБ

Спецслужбы обнаружили действующего иностранного агента. Шпiona то есть. Установили наблюдение и выяснили, что каждую неделю он через Интернет посылает кому-то странные нечитаемые тексты. Чтобы выяснить, к какой информации получил доступ шпион, требуется расшифровать информацию. Сотрудники спецслужб проникли в квартиру разведчика, изучили шифрующее устройство и выяснили принцип его работы.

На вход устройства подается строка текста $S_1 = s_1s_2...s_N$. Получив ее, устройство строит все циклические перестановки этой строки, то есть $S_2 = s_2s_3...s_Ns_1$, ..., $S_N = s_Ns_1s_2...s_{N-1}$. Затем множество строк $S_1, S_2, ..., S_N$ сортируется лексикографически по возрастанию. И в этом порядке строчки выписываются в столбец, одна под другой. Получается таблица размером $N \times N$. В какой-то строке K этой таблицы находится исходное слово. Номер этой строки вместе с последним столбцом устройство и выдает на выход.

Например, если исходное слово $S_1 = abracadabra$, то таблица имеет такой вид:

- 1. aabracadabr = S_{11}
- 2. abraabracad = S_8
- 3. abracadabra = S_1
- 4. acadabraabr = S_4
- 5. adabraabra = S_6
- 6. braabracada = S_9
- 7. bracadabra = S_2
- 8. cadabraabra = S_3
- 9. dabraabrac = S_7
- 10. raabracadab = S_{10}
- 11. racadabraab = S_5

И результатом работы устройства является число 3 и строка rdarcaaaabb.

Это все, что известно про шифрующее устройство. А вот дешифрующего устройства не нашли. Но поскольку заведомо известно, что декодировать информацию можно (а иначе зачем же ее передавать?), Вам предложили помочь в борьбе с хищениями секретов и придумать алгоритм для дешифровки сообщений. А заодно и реализовать дешифратор.

Исходные данные

В первой и второй строках находятся соответственно целое число и строка, возвращаемые шифратором. Длина строки и число не превосходят 100000. Строка содержит лишь следующие символы: a-z, A-Z, символ подчеркивания. Других символов в строке нет. Лексикографический порядок на множестве слов задается таким порядком символов:

ABCDEFGHIJKLMNOPQRSTUVWXYZ_abcdefghijklmnopqrstuvwxyz

Символы здесь выписаны в порядке возрастания.

Результат

Выведите декодированное сообщение в единственной строке.

Пример

исходные данные	результат
3 rdarcaaaabb	abracadabra

Автор задач: Идея — Александр Клепнин, подготовка — Александр Клепнин, Станислав Васильев

Источник задач: VIII Командный студенческий чемпионат Урала по программированию. Екатеринбург, 11-16 марта 2004 г.

Рис. 3. Условие задачи 1322.

3.1 Основная идея

Задача сводится к работе с преобразованием Барроуза–Уилера, в нашем случае необходимо выполнить обратное преобразование.

3.2 Краткое описание алгоритма

1. **Входные данные:** целое число s ($s \leq 100000$) и строка str .
2. Последовательно восстанавливаем строку. Пусть изначально известно, каким по порядку является приписанный в начало символ (его порядок в столбце)
3. Из предыдущего шага известно, какое место занимала строка без первого символа (i-ое).
4. Несложно заметить, что при выполнении такой операции строка с номером i всегда будет перемещаться на позицию с номером j .
5. **Выходные данные:** строка.


3.3 Листинг

Листинг 2. Исходный код для 1322

```
1 #include <iostream>
2
3
4 const int maximum_n = 100000;
5
6 std::pair<char, int> data[maximum_n];
7
8 bool comparator(std::pair<char, int> first, std::pair<char,
9     int> second) {
10     return (first.first != second.first) ? first.first <
11         second.first : first.second < second.second;
12 }
13
14 //sort in lexicographic order (quick sort)
15 void sorter(int left, int right) {
16     int i = left;
17     int j = right;
18
19     std::pair<char, int> cur = data[(left + right) / 2];
20
21     while (i <= j) {
22         while (comparator(data[i], cur)) {
23             ++i;
24         }
25         while (comparator(cur, data[j])) {
26             --j;
27         }
28     }
```

```
25         }
26
27         if (i <= j) {
28             std::swap(data[i++], data[j--]);
29         }
30     }
31
32     if (i < right) sorter(i, right);
33
34     if (j > left) sorter(left, j);
35 }
36
37
38 int main() {
39     int s;
40     std::string str;
41
42     std::cin >> s >> str;
43
44     int symbols_counter = str.length();
45
46     for (int i = 0; i < symbols_counter; ++i) {
47         data[i].first = str[i];
48         data[i].second = i;
49     }
50
51     sorter(0, symbols_counter - 1);
52
53     int result[symbols_counter];
54
55     for (int i = 0; i < symbols_counter; ++i) {
56         result[i] = data[i].second;
57     }
58
59     int current = s - 1;
60
61     for (int i = 0; i < symbols_counter; ++i) {
62         std::cout << data[current].first;
63         current = result[current];
64     }
65
66     return 0;
67 }
```


3.4 Результат



[Online Judge](#)
[О системе](#)
[Часто задаваемые вопросы](#)
[Новости сайта](#)
[Форум](#)
[Ссылки](#)

Задачи

[Архив задач](#)
[Отправить на проверку](#)
[Состояние проверки](#)
[Руководство](#)

Авторы

[Регистрация](#)
[Исправить данные](#)
[Рейтинг авторов](#)

Соревнования

[Текущее соревнование](#)
[Расписание](#)
[Прошлое соревнования](#)
[Правила](#)

Результаты проверки решений

ID	Дата	Автор	Задача	Язык	Результат проверки	№ теста	Время работы	Выделено памяти
10562333	19:24:35 21 фев 2024	miniDragon	1322. Шипы	Clang++ 17 x64	Accepted		0.031	1 612 KB

Рис. 4. Результат отправки задачи 1322.

4 Задача 1444

1444. Накормить элѣфпотама

Ограничение времени: 0.5 секунды

Ограничение памяти: 64 МБ

Гарри Поттер сдает экзамен по предмету «Уход за магическими существами». Его задание — накормить карликового элѣфпотама. Гарри помнит, что элѣфпотамы отличаются прямолинейностью и невозмутимостью. Они настолько прямолинейны, что ходят строго по прямой, и настолько невозмутимы, что заставить их идти можно, только если привлечь его внимание к чему-нибудь действительно вкусному. И главное, наткнувшись на цепочку своих собственных следов, элѣфпотам попадает в ступор и отказывается идти куда-либо. По словам Хагрида, элѣфпотамы обычно возвращаются домой, идя в обратную сторону по своим собственным следам. Поэтому они никогда не пересекают их, иначе могут заблудиться. Увидев свои следы, элѣфпотам детально вспоминает все свои перемещения от выхода из дома (поэтому-то они и ходят только по прямой и лишний раз не меняют направление — так легко запоминать). По этой информации элѣфпотам вычисляет, в какой стороне расположена его нора, после чего поворачивается и идет прямо к ней. Эти вычисления занимают у элѣфпотама некоторое (довольно большое) время. А то, что некоторые невежды принимают за ступор, на самом деле есть проявление выдающихся вычислительных способностей этого чудесного, хотя и медленно соображающего животного!

Любимое лакомство элѣфпотамов — слонови тывки, именно они и растут на лужайке, где Гарри должен сдавать экзамен. Перед началом испытания Хагрид притащит животное к одной из тывк. Скормив элѣфпотаму очередную тывку, Гарри может направить его в сторону любой оставшейся тывки. Чтобы сдать экзамен, надо провести элѣфпотама по лужайке так, чтобы тот съел как можно больше тывк до того, как наткнется на свои следы.

Исходные данные

В первой строке входа находится число N ($3 \leq N \leq 30000$) — количество тывк на лужайке. Тывки пронумерованы от 1 до N , причем номер один присвоен той тывке, у которой будет стоять элѣфпотам в начале экзамена. В следующих N строках даны координаты всех тывк по порядку. Все координаты — целые числа от -1000 до 1000 . Известно, что положения всех тывк различны, и не существует прямой, проходящей сразу через все тывки.

Результат

В первой строке выхода вы должны вывести K — максимальное количество тывк, которое может съесть элѣфпотам. Далее по одному числу в строке выведите K чисел — номера тывк в порядке их обхода. Первым в этой последовательности всегда должно быть число 1.

Пример

исходные данные	результат
4	4
0 0	1
10 10	3
0 10	2
10 0	4

Автор задачи: Идея и текст: Екатерина Васильева, программирование: Александр Мироненко, Алексей Лахтин, Ден Расквалов

Источник задачи: Х командный Чемпионат Урала по спортивному программированию, 24-25 марта 2006 года

Рис. 5. Условие задачи 1444.

4.1 Основная идея

4.2 Краткое описание алгоритма

1. Входные данные: в первой строке находится целое число N ($3 \leq N \leq 30000$) – количество тыкв на лужайке. Тыквы пронумерованы от 1 до N , причем номер один присвоен той тыкве, у которой будет стоять эфепотам в начале экзамена. В следующих N строках даны координаты всех тыкв по порядку. Все координаты – целые числа от -1000 до 1000 . Известно, что положения всех тыкв различны, и не существует прямой, проходящей через все тыквы сразу.

2.

3.

4.

5. Выходные данные: В первой строке выхода вывести K – максимальное количество тыкв, которое может съесть эфепотам. Далее по одному числу в строке вывести K чисел – номера тыкв в порядке их обхода. Первый в этой последовательности всегда должно быть число 1.

4.3 Листинг

Листинг 3. Исходный код для 1444

```
1 #include <iostream>
```

4.4 Результат

5 Вывод по работе

В ходе выполнения данной лабораторной работы были реализованы алгоритмы для решения задач 1207, 1322 и 1444.