Задание 4 “RMQ и деревья по неявному ключу”

# Задача 1. Вторая статистика (RMQ) (5 баллов)

Дано число N и последовательность из N целых чисел. Найти вторую порядковую статистику на заданных диапазонах.

Для решения задачи используйте структуру данных **Sparse Table.** Требуемое время обработки каждого диапазона O(1). Время подготовки структуры данных O(n log n).

Формат входных данных.

* В первой строке заданы 2 числа: размер последовательности N и количество диапазонов M.
* Следующие N целых чисел задают последовательность.
* Далее вводятся M пар чисел - границ диапазонов.

Гарантируется, что каждый диапазон содержит как минимум 2 элемента.

Формат выходных данных.

Для каждого из M диапазонов напечатать элемент последовательности - 2ю порядковую статистику. По одному числу в строке.

|  |  |
| --- | --- |
| in | out |
| 10 3  1 2 3 4 5 6 7 8 9 10  1 2  1 10  2 7 | 2  2  3 |

# Задача 2. Дерево отрезков (5 баллов)

**2\_1. Сумма на подотрезке.**

Дан массив из целых чисел *a1*, *a2*, ..., *an* (индексация с 1!). Для каждого запроса *[left, right]* найдите такой подотрезок *al*, *al+1*, ..., *ar  этого* массива (*1 <= left <= l <= r <= right <= n*), что сумма чисел *al* + *al+1* + ... + *ar* является максимально возможной.

Требуемое время ответа на запрос - O(log n).

Формат входных данных.

Входные данные содержат один или несколько тестовых примеров. Описание каждого из них начинается с двух чисел *n* и *m -* длины массива и числа интересующих подотрезков.

В следующей строке содержится *n* чисел — элементы массива. Каждое из этих чисел по абсо­лютной величине не превосходит 104.

Далее следуют описания подотрезков, каждое описание состоит из двух чисел *left* и *right* , обозначаю­щих левый и правый конец подотрезка (1 <= *left* *<= right <= n).*

Суммарная длина всех массивов, а также суммарное число подотрезков не превосходит 105.

Формат выходных данных.

Для каждого из тестовых примеров выведите *m* чисел: искомую максимальную сумму для каж­дого из подотрезков.

|  |  |
| --- | --- |
| in | out |
| 10 3  -100 1 2 3 4 -10 50 -100 -1 2  1 10  1 5  9 9  5 2  -1 2-1 2 -1  1 5  2 4 | 50  10  -1  3  3 |
|
|
|
|
|
|
|
|

**2\_2. Последовательность единиц.** Дан массив из нулей и единиц *a0*, *a1*, ..., *an-1*. Для каждого запроса *[left, right]* найдите такой подотрезок *al*, *al+1*, ..., *ar  этого* массива (*0 <= left <= l <= r <= right < n*), что числа *al*, *al+1*, …, *ar* являются максимально возможной последовательностью единиц.

Требуемое время ответа на запрос - O(log n).

Формат входных данных.

Описание каждого теста начинается с двух чисел *n* и *m -* длины массива и числа интересующих подотрезков.

В следующей строке содержится *n* нулей и единиц.

Далее следуют описания подотрезков, каждое описание состоит из двух чисел *left* и *right*, обозначаю­щих левый и правый конец подотрезка (0 <= *left <= right < n).*

Формат выходных данных.

Для каждого примера выведите *m* чисел: искомую максимальную длину последовательности единиц для каж­дого из подотрезков.

|  |  |
| --- | --- |
| in | out |
| 10 4  0 1 0 1 1 1 1 0 1 1  2 2  1 5  0 9  9 9 | 0  3  4  1 |
|
|
|
|
|
|
|
|

# Задача 3. LCA (5 баллов)

Задано дерево с корнем, содержащее 𝑛 (1 ≤ 𝑛 ≤ 100 000) вершин, пронумерованных от 0 до 𝑛−1. Требуется ответить на 𝑚 (1 ≤ 𝑚 ≤ 10 000 000) запросов о наименьшем общем предке для пары вершин. Запросы генерируются следующим образом. Заданы числа 𝑎1, 𝑎2 и числа 𝑥, 𝑦 и 𝑧. Числа 𝑎3, . . . , 𝑎2𝑚 генерируются следующим образом: 𝑎𝑖 = (𝑥·𝑎𝑖−2+𝑦·𝑎𝑖−1+𝑧) mod 𝑛. Первый запрос имеет вид ⟨𝑎1, 𝑎2⟩. Если ответ на 𝑖−1-й запрос равен 𝑣, то 𝑖-й запрос имеет вид ⟨(𝑎2𝑖−1 + 𝑣) mod 𝑛, 𝑎2𝑖⟩.

Для решения задачи можно использовать метод двоичного подъёма.

Формат входных данных.

* Первая строка содержит два числа: 𝑛 и 𝑚. Корень дерева имеет номер 0.
* Вторая строка содержит 𝑛 − 1 целых чисел, 𝑖-е из этих чисел равно номеру родителя вершины 𝑖.
* Третья строка содержит два целых числа в диапазоне от 0 до 𝑛−1: 𝑎1 и 𝑎2.
* Четвертая строка содержит три целых числа: 𝑥, 𝑦 и 𝑧, эти числа неотрицательны и не превосходят 109.

Формат выходных данных.

Выведите в выходной файл сумму номеров вершин — ответов на все запросы.

|  |  |
| --- | --- |
| in | out |
| 3 2  0 1  2 1  1 1 0 | 2 |

# 

# Задача 4. Супер-массив строк (5 баллов)

Реализуйте структуру данных “массив строк” на основе декартового дерева по неявному ключу со следующими методами:

// Добавление строки в позицию position.

// Все последующие строки сдвигаются на одну позицию вперед.

void InsertAt( int position, const std::string& value );

// Удаление строки из позиции position.

// Все последующие строки сдвигаются на одну позицию назад.

void DeleteAt( int position );

// Получение строки из позиции position.

std::string GetAt( int position );

Все методы должны работать за O(log n) в среднем, где n – текущее количество строк в массиве.

Формат входных данных.

Первая строка содержит количество команд *k* ≤ 106.

Последующие *k* строк содержат описания команд:

* Команда "+ 10 hello" означает добавление строки hello в позицию 10.
* Команда “- 14 16” означает удаление строк от позиции 14 до позиции 16 включительно.
* Команда “? 33” означает запрос на вывод строки из массива в позиции 33.

Формат выходных данных.

Выведите все строки, запрошенные командами “?”.

|  |  |
| --- | --- |
| in | out |
| 6  + 0 myau  + 0 krya  ? 0  + 2 gav  - 1 1  ? 1 | krya  gav |