**Задачи третьего модуля**

Контест для задачи 1: <http://tp-test1.corp.mail.ru/cgi-bin/new-client?contest_id=12>.

Контест для задачи 2: <http://tp-test1.corp.mail.ru/cgi-bin/new-client?contest_id=13>.

Контест для задач 3 и 4: <http://tp-test1.corp.mail.ru/cgi-bin/new-client?contest_id=15>.

**Задача 1. Порядок обхода.**

Дано число N < 106 и последовательность целых чисел из [-231..231] длиной N.

Требуется построить бинарное дерево, заданное наивным порядком вставки.

Т.е., при добавлении очередного числа K в дерево с корнем root, если root→Key ≤ K, то узел K добавляется в правое поддерево root; иначе в левое поддерево root.

**1\_1. Выведите элементы в порядке in-order (слева направо).**

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

**1\_2. Выведите элементы в порядке pre-order (сверху вниз).**

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

**1\_3. Выведите элементы в порядке post-order (снизу вверх).**

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

**1\_4. Выведите элементы в порядке level-order (по слоям, “в ширину”).**

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

**Задача 2. Декартово дерево.**

Дано число N < 106 и последовательность пар целых чисел из [-231..231] длиной N.

Построить декартово дерево из N узлов, характеризующихся парами чисел {Xi, Yi}.

Каждая пара чисел {Xi, Yi} определяет ключ Xi и приоритет Yi в декартовом дереве.

Построить также наивное дерево поиска по ключам Xi методом из задачи 1.

**2\_1.** Вычислить разницу глубин наивного дерева поиска и декартового дерева.

|  |  |
| --- | --- |
| in | out |
| 10  5 11  18 8  25 7  50 12  30 30  15 15  20 10  22 5  40 20  45 9 | 2 |
| 10  38 19  37 5  47 15  35 0  12 3  0 42  31 37  21 45  30 26  41 6 | 2 |

**2\_2.** Вычислить количество узлов в самом широком слое декартового дерева и количество узлов в самом широком слое наивного дерева поиска. Вывести их разницу.

|  |  |
| --- | --- |
| in | out |
| 10  5 11  18 8  25 7  50 12  30 30  15 15  20 10  22 5  40 20  45 9 | 1 |
| 10  38 19  37 5  47 15  35 0  12 3  0 42  31 37  21 45  30 26  41 6 | 1 |

**Задача 3. АВЛ-дерево.**

Дописать АВЛ-дерево. Методы повороты и удаление.

Дана последовательность команд добавления или удаления натуральных чисел в АВЛ-дерево. Команда добавления числа A задается положительным числом A, команда удаления числа A задается отрицательным числом “-A”. Требуется вывести высоту АВЛ-дерева после выполнения всех команд.

|  |  |
| --- | --- |
| in | out |
| 2 4 6 -2 | 2 |

**Задача 4. Использование АВЛ-дерева.**

**4\_1. Солдаты.** В одной военной части решили построить в одну шеренгу по росту. Т.к. часть была далеко не образцовая, то солдаты часто приходили не вовремя, а то их и вовсе приходилось выгонять из шеренги за плохо начищенные сапоги. Однако солдаты в процессе прихода и ухода должны были всегда быть выстроены по росту – сначала самые высокие, а в конце – самые низкие. За расстановку солдат отвечал прапорщик, который заметил интересную особенность – все солдаты в части разного роста. Ваша задача состоит в том, чтобы помочь прапорщику правильно расставлять солдат, а именно для каждого приходящего солдата указывать, перед каким солдатом в строе он должен становится. Требуемая скорость выполнения команды - O(log n).

Формат входных данных.

Первая строка содержит число N – количество команд (1 ≤ N ≤ 30 000). В каждой следующей строке содержится описание команды: число 1 и X если солдат приходит в строй (X – рост солдата, натуральное число до 100 000 включительно) и число 2 и Y если солдата, стоящим в строе на месте Y надо удалить из строя. Солдаты в строе нумеруются с нуля.

Формат выходных данных.

На каждую команду 1 (добавление в строй) вы должны выводить число K – номер позиции, на которую должен встать этот солдат (все стоящие за ним двигаются назад).

|  |  |
| --- | --- |
| in | out |
| 5  1 100  1 200  1 50  2 1  1 150 | 0  0  2  1 |

**4\_2. Порядковые статистики.** Дана число N и N строк. Каждая строка содержащит команду добавления или удаления натуральных чисел, а также запрос на получение k-ой порядковой статистики. Команда добавления числа A задается положительным числом A, команда удаления числа A задается отрицательным числом “-A”. Запрос на получение k-ой порядковой статистики задается числом k. Требуемая скорость выполнения запроса - O(log n).

|  |  |
| --- | --- |
| in | out |
| 5  40 0  10 1  4 1  -10 0  50 2 | 40  40  10  4  50 |

**Задача 5. Sparse Table.**

**5\_0.** Дано число N и последовательность из N целых чисел. Найти вторую порядковую статистику на заданном диапазоне.

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.

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

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

**Задача 6. Дерево отрезков.**

**6\_1. Система бронирования.** ООО Новые Железные Дороги поручило вам разработать систему бронирования билетов на новый маршрут поезда дальнего следования. Маршрут проходит через N станций, занумерованных от 0 до N-1. Вместимость поезда ограничена. В систему бронирования последовательно приходят запросы от пассажиров с указанием номера начальной и конечной станции, а также количество билетов, которые пассажир хочет приобрести. Требуемая скорость обработки каждого запроса - O(log N).

Формат входных данных.

Первая строка содержит число N – количество станций (1 ≤ N ≤ 50 000). Вторая строка содержит начальное количество проданных билетов. В примере продано по одному билету на перегонах 0->1 и 4->5. В третьей строке указана вместимость поезда. В четвёртой - количество запросов на бронирование. Далее идут непосредственно запросы в виде: номер начальной станции, номер конечной станции, количество запрашиваемых билетов.

Формат выходных данных.

Необходимо вывести номера запросов, которые не удалось выполнить из-за отсутствия свободных мест.

|  |  |
| --- | --- |
| in | out |
| 6  1 0 0 0 1  5  8  0 2 1  1 4 2  2 5 1  1 3 3  3 5 2  0 4 1  0 1 3  1 3 2 | 3 5 |

**6\_2. Художник и фотограф.** У художника-авангардиста есть полоска разноцветного холста. За один раз он перекрашивает некоторый отрезок полоски в некоторый цвет. После каждого перекрашивания специально обученный фотограф делает снимок части получившегося творения для музея современного искусства. Для правильного экспонирования требуется знать яркость самого темного цвета на выбранном фотографом отрезке. Напишите программу для определения яркости самого темного цвета на отрезке. Требуемая скорость определения яркости - O(log N).  
Цвет задается тремя числами R, G и B (0 ≤ R, G, B ≤ 255), его яркость = R + G + B. Цвет (R1, G1, B1) темнее цвета (R2, G2, B2), если R1 + G1 + B1 < R2 + G2 + B2.

Формат входных данных.

Первая строка содержит число N – длина отрезка (координаты [0...N-1]). Затем следует N строк по 3 числа – цвета отрезка.  
Следующая строка содержит число K – количество перекрашиваний и фотографирований. Затем задано K строк, описывающих перекрашивание и фотографирование. В каждой такой строке 7 натуральных чисел: C, D, R, G, B, E, F, где [C, D] – координаты перекрашиваемого отрезка, (R, G, B) – цвет покраски, [E, F] – фотографируемый отрезок.

Формат выходных данных.

Требуется вывести K строк. В каждой строке – яркость самого темного цвета на отрезке [E, F] после соответствующего покрашивания.

|  |  |
| --- | --- |
| in | out |
| 5  7 40 3  54 90 255  44 230 8  33 57 132  17 8 5  2  0 3 100 40 41 2 4  2 4 0 200 57 1 3 | 30  181 |