АиСД у2024. Второй семестр

Домашние задания М3134-М3135

⟨Версия от 28 апреля 2025 г.⟩

Темы

Неделя 1. Дерево отрезков

В заданиях с 1.1 по 1.6 дан массив a длины n. Требуется придумать, как при помощи дерева отрезков выполнять две операции. Первая операция — присвоить элементу a_i значение x. Вторая операция описана в каждом задании. Обе операции должны работать за $\mathcal{O}(\log n)$.

1.1. Найти минимум на отрезке $[l, r)$, а также вычислить количество элементов, равных минимуму.

1.2. Найти минимум на отрезке $[l, r)$, а также найти позицию самого левого элемента отрезка, который равен минимуму.

1.3. Найти значение суммы $a_l - a_{l+1} + a_{l+2} - \ldots + (-1)^{r-l} a_{r-1}$.

1.4. Найти значение суммы $a_l + 2a_{l+1} + 3a_{l+2} + \ldots + (r-l)a_{r-1}$.

1.5. Найти подотрезок $[l_1, r_1)$, такой что $l \le l_1 \le r_1 \le r$ и сумма на подотрезке $[l_1, r_1)$ максимальна среди всех таких отрезков. Достаточно найти значение самой суммы, хотя восстановить отрезок также не составит труда.

1.6. Найти минимальное i $(1 \le i \le n)$, такое что $a_i \ge k$. Здесь k — параметр, который задается в запросе. То есть, в разных запросах значения k могут различаться.

1.7. Дан массив из 0 и 1. Нужно найти количество непрерывных отрезков из единиц и уметь менять элемент на противоположный с помощью ДО

1.8. Дан массив из 0 и 1. Нужно Найти самый длинный непрерывный отрезок из единиц и уметь менять элемент на противоположный с помощью ДО

1.9. Научитесь искать НВП массива длины n за $\mathcal{O}(n \log n)$, используя дерево отрезков. Считайте, что элементы массива — натуральные числа, не превосходящие n.

1.10. Решите задачу 1.9, при условии, что элементы массива — произвольные целые числа.

1.11. Вычислите количество инверсий в массиве длины n за $\mathcal{O}(n \log n)$, используя дерево отрезков.

1.12. Дана строка из n открывающих и закрывающих круглых скобок. Придумайте, как при помощи дерева отрезков отвечать на следующие запросы за $\mathcal{O}(\log n)$. Первый запрос — изменить i-ю скобку. Второй запрос — проверить, является ли скобочная последовательность $a_l a_{l+1} \ldots a_r$ правильной.

1.13. Дана строка из n открывающих и закрывающих круглых скобок. Придумайте, как при помощи дерева отрезков отвечать на следующие запросы за $\mathcal{O}(\log n)$. Первый запрос — изменить i-ю скобку. Второй запрос — найти длину наибольшего префикса отрезка $[l, r)$, который является правильной скобочной последовательностью.

1.14. Дан массив длины n, элементы которого являются натуральными числами, не превосходящими n. Научитесь отвечать на запрос: даны l, r, x и y, требуется вычислить количество элементов на отрезке $[l, r)$, которые лежат в диапазоне от x до y (то есть количество таких i, что $l \le i < r$ и $x \le a_i \le y$). В данной задаче считайте, что все запросы известны заранее, то есть можно решать задачу в Offline. Время работы: $\mathcal{O}((n+q)\log n)$.

1.15. Дан массив длины n, элементы которого являются натуральными числами, не превосходящими n. Научитесь отвечать на запрос: даны l, r, требуется вычислить количество различных элементов, которые встречаются на отрезке $[l, r)$. В данной задаче считайте, что все запросы известны заранее, то есть можно решать задачу в Offline. Время работы: $\mathcal{O}((n+q)\log n)$.

1.16. Нужно реализовать две операции:

(a) Определить значение на позиции pos.

(b) Увеличить числа с l-й до r-й на величину d.

1.17. Марио собирается проходить уровень, состоящий из n последовательно расположенных труб, высота i-й трубы — a_i. Он может переместиться с трубы i на трубу j, если $|i-j| = 1$ и $a_j - a_i \le 1$. Требуется выполнять операции двух типов за $\mathcal{O}(\log n)$:

(a) Определить, может ли Марио добраться от трубы с номером x до трубы с номером y.

(b) Увеличить высоты труб с l-й до r-й на величину d.

Неделя 2. Дерево отрезков — 2

В заданиях с 18 по 24 дан массив a длины n. Требуется придумать, как при помощи дерева отрезков выполнять указанные операции за $\mathcal{O}(\log n)$. Обратите внимание, при выполнении заданий следует уделить особое внимание псевдокоду функций проталкивания отложенных операций `push()`.

2.18. (a) Присвоить значение x всем элементам отрезка
(b) Умножить все элементы отрезка на -1 (то есть заменить a_i на $-a_i$)
(c) Найти сумму на отрезке

2.19. (a) Присвоить значение x всем элементам отрезка
(b) Умножить все элементы отрезка на -1 (то есть заменить a_i на $-a_i$)
(c) Найти максимум на отрезке

2.20. (a) Присвоить значение x всем элементам отрезка
(b) Умножить все элементы отрезка на -1 (то есть заменить a_i на $-a_i$)
(c) Найти подотрезок с максимальной суммой

2.21. (a) Присвоить значение x всем элементам отрезка
(b) Прибавить значение x ко всем элементам отрезка
(c) Найти значение элемента

2.22. (a) Присвоить значение x всем элементам отрезка
(b) Прибавить значение x ко всем элементам отрезка
(c) Найти сумму на отрезке

2.23. (a) Заменить на отрезке a_i на $\max(a_i, x)$
(b) Заменить на отрезке a_i на $\min(a_i, x)$
(c) Найти значение элемента

2.24. (a) Присвоить значение x всем элементам отрезка
(b) Найти подотрезок максимальной длины, состоящий из одинаковых чисел

В заданиях с 2.25 по 2.28 дан массив a длины n, состоящий из булевых значений. Требуется придумать, как при помощи дерева отрезков выполнять указанные операции за $\mathcal{O}(\log n)$.

2.25. (a) Присвоить значение x всем элементам отрезка
(b) Найти ближайшую к i-му элементу единицу

2.26. (a) Изменить все значения на отрезке на противоположные
(b) Найти количество единиц на отрезке

2.27. (a) Присвоить значение x всем элементам отрезка
(b) Найти количество непрерывных отрезков из единиц

2.28. (a) Присвоить значение x всем элементам отрезка

(b) Найти самый длинный непрерывный отрезок из единиц

2.29. Дан массив длины n. Вычислите количество возрастающих подпоследовательностей массива длины k. Время $\mathcal{O}(kn \log n)$.

2.30. Петя едет из Питера в Москву на машине. По пути ему встретятся n заправок, для каждой известно ее положение и стоимость литра бензина. Также известно, сколько бензина тратит машина на километр пути и сколько бензина помещается в бак. Нужно доехать, потратив минимальную сумму.

2.31. Есть n домашних заданий, которые нужно сделать, для каждого дела известно, сколько времени нужно на него потратить t_i и до какого времени его нужно сделать d_i. Кроме того, для каждого дз известно, в какое время пришлют задание s_i (соответственно, раньше начать его делать не получится). Составить план работы так, чтобы успеть все сделать вовремя, если можно переключаться с задания на задание (то есть например сделать частично первое, переключиться на второе, потом доделать первое,...).

Неделя 3. Сортировка событий

3.32. Есть n касс. Каждая касса работает без перерыва определенный промежуток времени по фиксированному расписанию (одному и тому же каждый день). Требуется определить, на протяжении какого времени в течение суток работают все кассы одновременно.

3.33. Задачи участникам раздаются последовательно, а не все в самом начале тура, и каждая i-я задача становится доступной участникам в свой момент времени s_i. При поступлении очередной задачи каждый участник должен сразу определить, будет он ее решать или нет. В случае, если он выбирает для решения эту задачу, то у него есть t_i минут на то, чтобы сдать ее решение на проверку, причем в течение этого времени он не может переключиться на решение другой задачи. Если же участник отказывается от решения этой задачи, то в будущем он не может к ней вернуться. В тот момент, когда закончилось время, отведенное на задачу, которую решает участник, он может начать решать другую задачу, ставшую доступной в этот же момент, если такая задача есть, или ждать появления другой залачи. При этом за правильное решение i-й задачи участник получает c_i баллов. **В этой задаче все c_i одинаковые**

3.34. В этой задаче все c_i разные

3.35. В этом году погода стоит особо жаркая, поэтому в комповнике очень душно и важно следить за тем, чтобы в комповнике не находилось одновременно очень много школьников. Поэтому завуч записал время прихода и ухода из комповника каждого ЛКШонка.

Теперь завуч хочет узнать, сколько ЛКШат встретил в комповнике каждый ЛКШонок.

3.36. На прямой задано некоторое множество отрезков с целочисленными координатами концов $[L_i, R_i]$. Выберите среди данного множества подмножество отрезков, целиком покрывающее отрезок $[0, M]$

3.37. Ученые планируют набор продуктов для экспедиции на Марс. Планируется, что запас экспедиции будет состоять из n типов продуктов. У экспедиции будет k_i порций продуктов i-го типа. Продукт i-го типа должен быть использован на протяжении t_i дней после начала экспедиции, после чего портится. Если за t_i дней не все порции продукты i-го типа съедены, то все оставшиеся порции этого продукта уничтожаются.

В экспедицию планируют направить c участников. Каждый день участники экспедиции выбирают любые c имеющихся у них порций и съедают их. Разные участники экспедиции могут есть как одинаковые, так и различные типы продуктов.

Отдел планирования снабжения хочет понять, насколько избыточен набор продуктов, запланированный для экспедиции. Они хотят выяснить, какое максимальное различное количество типов продуктов участники экспедиции смогут полностью съесть в процессе экспедиции, не допустив уничтожения ни одной их порции продукта этого типа.

3.38. У Поликарпа есть шахматная доска размера $n \times m$, на которой расставлены k ладей. Поликарп еще не придумал правила игры, в которую он будет играть. Однако он уже выделил на доске q прямоугольных участков особой стратегической важности, которые должны быть надежно защищены. По мнению Поликарпа, прямоугольный участок доски надежно защищен, если все его свободные клетки бьются ладьями, стоящими на этом участке. Ладьи на остальной части доски на защиту участка не влияют. Расстановка ладей фиксирована и не может быть изменена. Напомним, что ладья бьет все клетки, расположенные с ней на одной вертикали или горизонтали, если между клеткой и ладьей нет других фигур. Помогите Поликарпу определить, все ли стратегически важные участки надежно защищены.

3.39. в течение дня для каждого покупателя, посетившего супермаркет, было зафиксировано время, когда он пришел в супермаркет, и когда он из него ушел. Менеджер по рекламе предположил, что такое расписание прихода-ухода покупателей сохранится и в последующие дни. Он хочет составить расписание трансляции рекламных роликов, чтобы каждый покупатель услышал не меньше двух рекламных объявлений. В тоже время он выдвинул условие, чтобы два рекламных объявления не транслировались одновременно и, поскольку продавцам все время приходится выслушивать эту рекламу, общее число рекламных объявлений за день было минимальным.

3.40. Всем известно, что в мировой истории произошло ровно n событий: i-ое событие продолжалось с a_i по b_i годы включительно ($a_i < b_i$). Событие j включает в себя событие i, если $a_j < a_i$ и $b_i < b_j$. Ваша же задача проще: найдите количество событий, которые включены в какое-либо другое событие.

3.41. Разберитесь в алгоритме поиска площади объединения прямоугольников для больших координат и расскажите как надо поменять алгоритм с лекции.

3.42. Найдите площадь объединения ромбов с координатами центра в точке (x_i, y_i) и уравнением $|x - x_i| + |y - y_i| \le d_i$.

3.43. В руках шефа каким-то образом оказались сведения о надвигающейся ревизии — список из N грузов, которые будут контролироваться Министерством. Для каждого груза известны время его прибытия T_i, отсчитываемое с некоторого момента, хранимого в большом секрете, и время L_i, требуемое аппарату для обработки этого груза. Шеф дал Мише задание по этим данным определить, какое минимальное количество аппаратов необходимо заказать на заводе, чтобы все грузы Министерства начинали досматриваться сразу после прибытия. Необходимо учесть, что конструкция тех аппаратов, которые было решено установить, не позволяет обрабатывать два груза одновременно на одном аппарате. Напишите программу, которая поможет Мише справиться с его задачей.

3.44. Найдите длину пересечения n отрезков.

3.45. Вам дано n прямоугольников. Для каждого прямоугольника определите длину пересечения всех прямоугольников, не включая его.

3.46. Экзамен по берляндскому языку проходит в узкой и длинной аудитории. На экзамен пришло N студентов. Все они посажены в ряд. Таким образом, позиция

каждого человека задается координатой на оси Ox (эта ось ведет вдоль длинной аудитории). Два человека могут разговаривать, если расстояние между ними меньше или равно D. Какое наименьшее количество типов билетов должен подготовить преподаватель, чтобы никакие два студента с одинаковыми билетами не могли разговаривать? Выведите способ раздачи преподавателем билетов.

Неделя 4. Двоичное дерево поиска. AVL-дерево

4.47. Придумайте, как реализовать рекурсивную функцию, выводящую все элементы двоичного дерева поиска в порядке сортировки, за $\mathcal{O}(n)$.

4.48. Придумайте, как реализовать нерекурсивную функцию, выводящую все элементы двоичного дерева поиска в порядке сортировки, за $\mathcal{O}(n)$, используя $\mathcal{O}(1)$ дополнительной памяти. Можно считать, что у каждой вершины хранится ссылка на родительскую вершину.

4.49. Докажите, что не существует алгоритма, который строит двоичное дерево поиска по заданному набору ключей быстрее, чем за $\Omega(n \log n)$ в худшем случае.

4.50. Научитесь поддерживать мультимножество.

4.51. Научитесь находить k-й в порядке возрастания ключ в двоичном дереве поиска за $\mathcal{O}(H)$. Разрешается хранить в каждой вершине $\mathcal{O}(1)$ дополнительной памяти.

4.52. Научитесь по данному ключу x находить количество элементов в дереве, не превосходящих x за $\mathcal{O}(H)$. Разрешается хранить в каждой вершине $\mathcal{O}(1)$ дополнительной памяти.

4.53. По заданной вершине в дереве найдите следующие k вершин в порядке сортировки за $\mathcal{O}(H + k)$.

4.54. Приведите пример дерева, в котором средняя глубина вершины (среднее расстояние от корня до вершины) $\mathcal{O}(\log n)$, а высота дерева (максимальное расстояние от корня до вершины) асимптотически больше, чем $\log n$.

4.55. Приведите пример AVL-дерева и операции добавления элемента, при которой операцию балансировки будет выполнена более, чем в одной вершине.

4.56. Приведите пример AVL-дерева и операции удаления элемента, при которой операция балансировки будет выполнена более, чем в одной вершине.

4.57. Приведите пример двух AVL-деревьев, хранящих одно множество ключей, но имеющих разную высоту.

4.58. Научитесь проверять, что заданное двоичное дерево является корректным двоичным деревом поиска.

4.59. Пусть в двоичном дереве поиска для каждой вершины высота ее детей отличается не более, чем на 5. Правда ли, что высота такого дерева $\mathcal{O}(\log n)$?

4.60. Пусть в двоичном дереве поиска для каждой вершины высота ее детей отличается не более, чем в два раза. Правда ли, что высота такого дерева $\mathcal{O}(\log n)$?

4.61. Научитесь при помощи AVL-дерева отвечать на следующие запросы за $\mathcal{O}(\log n)$:

(a) Добавить число x в множество

(b) Удалить число x из множества

(c) Вычислить сумму всех x из множества, таких что $l \leq x \leq r$

4.62. Рассмотрим два AVL-дерева. Пусть любой ключ в первом дереве меньше, чем любой ключ во втором дереве. Предложите способ слить эти два дерева в одно AVL-дерево за истинные $\mathcal{O}(\log n)$, где n — сумма размеров деревьев.

4.63. Модифицируйте AVL-дерево таким образом, чтобы в каждой вершине хранить, помимо ссылок на детей, $\mathcal{O}(1)$ бит дополнительной информации (обратите внимание, что стандартная реализация с хранением высот вершин использует больше памяти).

Неделя 5. Декартово дерево

5.64. Дан массив пар (x, y), отсортированный по возрастанию значений x. Постройте по нему декартово дерево за $\mathcal{O}(n)$.

5.65. Можно ли построить декартово дерево за $\mathcal{O}(n)$, если массив пар не отсортирован по x?

5.66. Дан массив пар (x, y). Все x попарно различны, а y могут совпадать. Как проверить, что декартово дерево можно построить единственным образом?

5.67. Дан массив пар (x, y). Все x попарно различны, а y могут совпадать. Необходимо вычислить количество способов построить декартово дерево.

5.68. Придумайте, как сделать СНМ на базе дерева поиска по неявному ключу с временем работы операций $\mathcal{O}(\log n)$.

5.69. Пусть у дерева поиска нет вершин с одним ребенком (то есть у всех вершин 0 или 2 детей). Правда ли, что высота такого дерева $\mathcal{O}(\log n)$?

5.70. Пусть у дерева поиска размер каждого поддерева равен $2^k - 1$ для некоторого целого k (для разных вершин могут быть разные значения k). Правда ли, что высота такого дерева $\mathcal{O}(\log n)$?

5.71. Дан массив чисел длины n. При помощи декартова дерева научитесь выполнять следующие операции за $\mathcal{O}(\log n)$:

(a) Развернуть отрезок массива $[l, r]$ задом наперед

(b) Найти минимум на отрезке $[l, r]$

5.72. Дан массив чисел длины n. При помощи декартова дерева научитесь выполнять следующие операции за $\mathcal{O}(\log n)$:

(a) Для данных l и r поменять местами следующие пары элементов: l и $l + 1$, $l + 2$ и $l + 3$, и так далее до r

(b) Найти минимум на отрезке $[l, r]$

5.73. Будем строить декартово дерево, не храня приоритеты. В тех местах, где производится сравнение приоритетов, будем выбирать случайный вариант с вероятностью $\frac{1}{2}$. Покажите, что при такой реализации декартова дерева некоторая последовательность операций может привести к тому, что высота дерева станет $\Omega(n)$.

5.74. В дереве поиска, в отличие от дерева отрезков, исходные элементы множества хранятся не только в листьях, но и в промежуточных узлах. Иногда это неудобно, поэтому делают другую версию дерева поиска: исходные элементы множества хранятся только в листьях, а в промежуточных вершинах хранится максимальный ключ в поддереве. Покажите, как осуществлять операции поиска и добавления элемента в таком дереве.

5.75. Дерево поиска вывели на экран при помощи такой функции:

```
print_tree(t):
    if (t == null):
        return
    print(t.key)
    print_tree(t.l)
    print_tree(t.r)
```

По массиву, который вывела данная процедура, восстановите дерево поиска за $\mathcal{O}(n)$.

5.76. Даны n окружностей. Они не пересекаются, но могут быть вложены друг в друга. Постройте дерево вложенности окружностей за $\mathcal{O}(n \log n)$.

Неделя 6. Splay-дерево

6.77. Докажите по аналогии с другими операциями, что операция `зиг-заг` имеет амортизированное время работы $\widetilde{T}(\texttt{зиг-зиг}(x)) = 3 \cdot (r'(x) - r(x))$.

6.78. Пусть была сделана операция `splay` для вершины x, находящейся на глубине h. Как изменится сумма глубин вершин на пути от вершины x до корня дерева?

6.79. Упростим операцию `splay`: будем всегда делать операцию `зиг`, забыв про существование `зиг-зиг` и `зиг-заг`. То есть, вне зависимости от ориентации ребер, всегда будем поворачивать поддерево родителя текущей вершины влево или вправо, чтобы поднять текущую вершину на 1 вверх. Покажите, что существует последовательность из n операций, которая в этом случае будет работать дольше, чем за $n \log n$.

6.80. Пусть в Splay дереве находятся числа от 1 до n. За какое суммарное время отработает последовательность операций: `find(1)`, `find(2)`, ..., `find(n)`?

6.81. Пусть в Splay дереве находятся числа от 1 до n. Будем делать m операций в следующей последовательности: `splay(1)`, `splay(n)`, `splay(1)`, `splay(n)`, ... За какое суммарное время они отработают?

6.82. Пусть в Splay дереве размера n вы совершаете много операций над небольшим подмножеством элементов размера k. Как будет выглядеть дерево? За какое время будут работать операции?

6.83. Пусть в Splay дереве находятся числа от 1 до n. Далее поступают m запросов, суммарное количество запросов к i-му элементу равно p_i. Докажите, что суммарное время всех запросов равно $\mathcal{O}(m + \sum p_i \log \frac{m}{p_i})$.

Подсказка: на лекции мы ввели $s(x)$ как количество вершин в поддереве вершины x. Теперь можно придумать для каждой вершины некоторый вес $w(x)$ (подумайте, какой) и обозначить за $s(x)$ сумму весов всех вершин в поддереве вершины x.

6.84. Придумайте способ построить Splay дерево за $\mathcal{O}(n)$ по заданному отсортированному массиву.

6.85. Пусть в Splay дереве находятся числа от 1 до n. Далее поступают m запросов к вершинам $x_1, x_2, \ldots, x_m$. При этом известно, что $|x_i - x_{i-1}| \le d$. Докажите, что суммарное время запросов равно $\mathcal{O}(m \log d)$.

6.86. Докажите, что если сделать операцию `splay` ко всем элементам в порядке возрастания ключа, то суммарное время работы будет $\mathcal{O}(n)$, вне зависимости от начальной структуры дерева.

6.87. Постройте пример Splay дерева из хотя бы семи вершин, которое после применения операции `splay` к одному из самых глубоких листьев становится бамбуком.

6.88. В пустое сплей-дерево были добавлены элементы в таком порядке: $30, 40, 60, 50, 80, 70$. Какой элемент нужно добавить, чтобы дерево имело минимальную глубину? А какой, чтобы максимальную? Нарисуйте получившиеся деревья.

6.89. Приведите пример, когда последовательная вставка и затем удаление одного и того же значения в AVL-дереве приводит к дереву другой формы.

6.90. Пусть задана структура AVL-дерева из n вершин. Покажите, что можно в каком-то порядке вставить n ключей в пустое дерево, используя стандартный алгоритм вставки в AVL-дерево, чтобы получить дерево именно такой структуры.

6.91. Придумайте способ добавить в Splay-дерево операцию `split_size(T, s)`, которая должна разделить дерево T на два дерева T_1 и T_2 таким образом, что любой ключ дерева T_1 меньше любого ключа дерева T_2, а количество ключей в дереве T_1 равно s. Амортизированное время работы всех операций не должно превышать $\mathcal{O}(\log n)$. Разрешено использовать $\mathcal{O}(1)$ дополнительной памяти в каждой вершине.

Неделя 7. Дерево Фенвика. Sparse Table

7.92. На лекции был рассмотрен способ построить дерево Фенвика за $\mathcal{O}(n)$. Научитесь строить дерево Фенвика за $\mathcal{O}(n)$ с использованием $\mathcal{O}(1)$ дополнительной памяти.

7.93. Дано построенное дерево Фенвика для некоторого массива. Восстановите по нему исходный массив, используя $\mathcal{O}(1)$ дополнительной памяти.

7.94. Как изменится время работы и необходимый объем памяти Sparse Table, если хранить значения функции на отрезках длины x^k, а не 2^k ($x > 2$)? Приведите оценки, зависящие от n и x.

7.95. Добавьте в дерево Фенвика операцию, позволяющую найти префикс максимальной длины, сумма на котором не превосходит x (все числа в массиве неотрицательные). Время работы: $\mathcal{O}(\log n)$.

7.96. Научитесь отвечать на запросы за $\mathcal{O}(\log n)$ с использованием дерева Фенвика:

(a) Прибавить x ко всем элементам отрезка $[l, r)$.

(b) Найти значение элемента a_i.

7.97. Научитесь отвечать на запросы за $\mathcal{O}(\log n)$ с использованием дерева Фенвика:

(a) Прибавить x ко всем элементам отрезка $[l, r)$.

(b) Найти сумму элементов на отрезке $[l, r)$.

7.98. Научитесь отвечать на запросы за $\mathcal{O}(\log n)$ с использованием дерева Фенвика:

(a) Найти минимум на отрезке $[0, r)$.

(b) Заменить i-й элемент на число d.

Можно ли решать данную задачу для произвольных значений d?

7.99. Научитесь обрабатывать следующие запросы при помощи дерева Фенвика за $\mathcal{O}(\log n)$:

(a) Присвоить значение d всем элементам отрезка $[i, n)$.

(b) Найти значение i-го элемента массива.

7.100. Даны массивы a и b длины n. Найдите количество отрезков $[l, r)$, таких что $\min(a_l, a_{l+1}, \ldots, a_{r-1}) = \max(b_l, b_{l+1}, \ldots, b_{r-1})$. Время работы: $\mathcal{O}(n \log n)$.

7.101. Дан массив a длины n. Найдите количество отрезков $[l, r)$, таких что $(a_l \text{ or } a_{l+1} \text{ or } \ldots \text{ or } a_{r-1}) > \max(a_l, a_{l+1}, \ldots, a_{r-1})$. Здесь or означает побитовое «ИЛИ». Время работы: $\mathcal{O}(n \log n \log C)$, где C — максимальное из a_i. Подсказка: используйте Sparse Table!

7.102. Рассмотрим следующий код альтернативной реализации дерева Фенвика:

```
sum(r):
    i = r + 1
    ans = 0
    while i > 0:
        ans += t[i]
        i -= i & -i
    return ans

add(pos, d):
    i = pos + 1
    while i <= n:
        t[i] += d
        i += i & -i
```

Здесь массив t имеет длину $n + 1$, а i — signed переменная. Докажите, что данные функции работают за $\mathcal{O}(\log n)$ и являются корректной реализацией дерева Фенвика.

7.103. Подумайте, безопасно ли писать код из предыдущего задания на языке C++?

7.104. Дана таблица размера $n \times n$. Обработайте следующие запросы за $\mathcal{O}(\log^2 n)$:

(a) Прибавить константу ко всем ячейкам подпрямоугольника.

(b) Найти значение суммы на подпрямоугольнике.

7.105. На очередных тренировочных сборах команд программистов состоялось три соревнования. Теперь каждая команда считает себя сильнее всех команд, которых она обыграла хотя бы на одном из этих соревнований. Сколько существует пар команд, в которых каждая команда считает себя сильнее другой? Время работы: $\mathcal{O}(n \log^3 n)$.

7.106. Есть n точек в трехмерном пространстве, координаты целые и не больше k. Найти последовательность точек максимальной длины такую, что точки в ней монотонно возрастают по всем трем координатам. Время работы: $\mathcal{O}(k^2 + n \log n \log^2 k)$.

7.107. Дана таблица $n \times n$, в некоторых клетках которой находятся фишки. Нужно по запросу (x, y, d) за $\mathcal{O}(\log^2 n)$ находить число фишек, находящихся на манхеттенском расстоянии не больше d от точки (x, y).

7.108. Есть шахматное поле $n \times n$. Требуется обрабатывать следующие запросы за $\mathcal{O}(\log n)$:

(a) Добавить ладью на поле

(b) Удалить ладью с поля

(c) Найти количество клеток, которые не бьет ни одна ладья

7.109. Дана таблица размера $n \times n$. Научитесь находить сумму чисел в прямоугольном треугольнике с углами в клетках (i, j), $(i + d, j)$, $(i, j + d)$. Разрешается сделать $\mathcal{O}(n^2)$ препроцессинга.

7.110. (a) Прибавить ко элементам отрезка арифметическую прогрессию (то есть к a_l прибавить b, к a_{l+1} — $b+k$, к a_{l+2} — $b+2k$, и так далее)

(b) Найти сумму на отрезке

Неделя 8. Внешняя память

8.111. Решение задачи List Ranking (1)

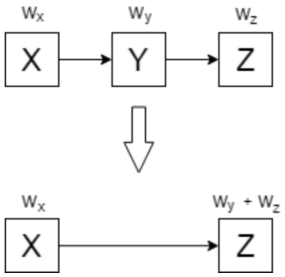
Есть 3 таблицы:

(a) Таблица $Conn$ из пар (i, j), где каждая пара значит, что после i-ого элемента идет j-ый (может быть получена из входных данных за время линейного сканирования)

(b) Таблица W из пар (i, w_i), хранящая веса элементов

(c) Таблица D, в которой записаны удаляемые элементы

Как получить новую таблицу R из пар (i, w'_i), хранящая новые веса элементов. Оцените сложность алгоритма



8.112. Решение задачи List Ranking (2)

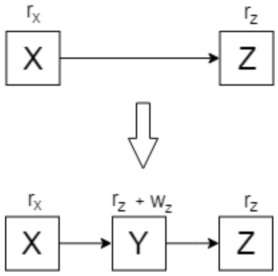
По возвращению из рекурсии нужно пересчитать веса удаленных элементов. Рассмотрим 3 таблицы:

(a) Таблица $RevConn$ из пар (j, i), где каждая пара значит, что после i-ого элемента идет j-ый

(b) Таблица W из пар (i, w_i), хранящая веса элементов

(c) Таблица R из пар (i, r_i), хранящая новые веса элементов модифицированного списка

Как получить таблицу со всеми весами элементов удаленных на этой итерации. Оцените сложность алгоритма



8.113. Решение задачи List Ranking (3)

Как выбрать удаляемые элементы? Оцените сложность алгоритма

8.114. Заданы массивы пар: (p_i, a_i) длины m_1 и (q_i, b_i) длины m_2. Значения p_i различны и от 1 до $m_1 + m_2$. Значения q_i так же различны и от 1 до $m_1 + m_2$. Научитесь составлять список пар (a_i, b_j), если $p_i = q_j$, $(a_i, 0)$, если для i такого j не существует, и $(0, b_j)$, если для j такого i не существует, за $\mathrm{Sort}(m_1 + m_2)$.

8.115. Во внешней памяти задана матрица $n \times n$ чисел. Транспонируйте матрицу за $\mathcal{O}\left(\left\lceil \frac{n^2}{B} \right\rceil\right)$. В этой задаче предполагайте, что у вас $M > B \cdot B$.

8.116. Предложите, как из матрицы a_{ij} размера $n \times n$ получить массив $b_{ij} = \sum\limits_{(x \le i) \wedge (y \le j)} a_{xy}$ за $\mathcal{O}\left(\left\lceil \frac{n^2}{B} \right\rceil\right)$.

8.117. Предложите, как решить задачу НОП за $\mathcal{O}\left(\left\lceil \frac{(n+m)^2}{B} \right\rceil\right)$.

8.118. Предложите, как решить задачу о рюкзаке за $\mathcal{O}\left(\left\lceil \frac{nW}{B} \right\rceil\right)$.

8.119. На жестком диске подряд записан массив из целых чисел размера n. Предложите способ ответить на k запросов поиска суммы на подотрезке массива. Вы можете выполнить предподсчёт за $\mathcal{O}\left(\frac{n}{B}\right)$, после чего время ответа на один запрос в модели внешней памяти должно составлять $\mathcal{O}(1)$, ответ на запрос в онлайне. Считайте, что $n > M$ и $k > M$.

8.120. Дан массив, найти k-ю порядковую статистику за $\mathcal{O}(\frac{N}{B})$.

8.121. Есть матрица $n \times n$. Найти путь из верхнего левого угла в правый нижний с максимальной суммой (ходить можно только вправо и вниз). Время $\mathcal{O}(\frac{n^2}{B})$ (найдите только вес пути, можете подумать как восстановить сам путь).

Неделя 9. 2-3 дерево и какие-то задачи

9.122. Докажите, что в подвешенном дереве с m листьями, в котором у каждой вершины (кроме листьев) два или более ребенка, $\mathcal{O}(m)$ вершин

9.123. Требуется ответить на n запросов в online за $\mathcal{O}(\log n)$:

- ▷ `insert x` — добавить число x;
- ▷ `remove x` — удалить число x;
- ▷ `getSum l r` — посчитать сумму всех x из структуры таких, что $l \le x \le r$;
- ▷ `getSumByTime l r` — посчитать сумму всех x из структуры таких, что x добавлен запросом с номером от l до r.

9.124. Разработайте структуру данных, которая поддерживает множество непересекающихся отрезков на прямой вида $[a, b)$, $0 \le a < b \le u$. Структура данных должна поддерживать следующие операции за время $\mathcal{O}(\log \log u)$:

- ▷ `insert`(a, b): добавить отрезок $[a, b)$ (если он не пересекается с уже добавленными отрезками).
- ▷ `unite`(a, b, c): слить отрезки $[a, b)$ и $[b, c)$ в отрезок $[a, c)$.
- ▷ `split`(a, b, c): разбить отрезок $[a, c)$ на отрезки $[a, b)$ и $[b, c)$.
- ▷ `find`(x): сообщить, какому отрезку принадлежит точка x.

9.125. Представим, что вы реализуете обычное дерево поиска, с наивным добавлением и удалением без балансировки. Предложите модификацию алгоритма хранения и операций с деревом, которые не изменяет структуру дерева (отношения ребенок-родитель, рисунок дерева должен оставаться таким же), который поможет реализовать функции `next(v)` и `prev(v)`, которые по вершине дерева возвращают вершину, в которой хранится следующий и предыдущий ключ, соответственно. Время работы `next` и `prev` истинное $\mathcal{O}(1)$, а остальных функций осталось неизменным. Вы можете воспользоваться $\mathcal{O}(1)$ дополнительной памятью в каждой вершине.

9.126. Докажите, что функция `veryNext` работает за $\mathcal{O}(\log n + k)$, если `next` — функция, которая находит вершину сбалансированного двоичного дерева поиска (т.е. двоичного дерева поиска высоты $\mathcal{O}(\log n)$ со следующим ключом

```
fun veryNext(start: Node, k: Int): Node {
  var v = start
  for (i in 1..k) {
    v = next(v)
  }
  return v
}
```

9.127. Предложите алгоритм слияния двух 2-3-деревьев, при том, что в первом дереве все ключи меньше, чем во втором за $\mathcal{O}(\log n)$.

9.128. Вам задано 2-3 дерево и ключ x. Предложите алгоритм разбиения заданного дерева на два: в первом должны оказаться все ключи меньшие либо равные x, а во втором — все ключи большие x. Алгоритм должен работать за $\mathcal{O}(\log n)$, где n — размер заданного дерева.

9.129. Вам известно, что в двоичном подвешенном дереве глубины листьев равны $d_1, d_2, \dots d_m$. Докажите, что $\sum\limits_{i=1}^{m} 2^{-d_i} \leq 1$. Сформулируйте критерий, при котором $\sum\limits_{i=1}^{m} 2^{-d_i} = 1$

9.130. Вам задано двоичное подвешенное дерево из n вершин. Вам также задано n различных целых чисел. Докажите, что существует ровно один способ расставить числа в вершины как ключи, чтобы получилось дерево поиска.

9.131. Для произвольного массива a длины n обозначим функцию от массива как $\sum_1^n a_i \cdot i$.

Научитесь за быстро поддерживать функцию от массива после каждой операции

- ▷ Выполнить циклический сдвиг массива. То есть, массив $[a_1, a_2, \dots, a_n]$ становится $[a_n, a_1, a_2, \dots, a_{n-1}]$
- ▷ Развернуть весь массив. То есть, массив $[a_1, a_2, \dots, a_n]$ становится $[a_n, a_{n-1}, \dots, a_1]$.
- ▷ Добавить элемент в конец массива. Массив $[a_1, a_2, \dots, a_n]$ становится $[a_1, a_2, \dots, a_n, k]$ после добавления k в конец массива.

9.132. Докажите, что любое двоичное дерево поиска T_1 может быть преобразовано в дерево T_2 с теми же ключами, что и T_1, используя малые повороты АВЛ-дерева. (Обратите внимание, что в этой задаче T_1 и T_2 не обязательно являются АВЛ-деревьями, а являются обычными деревьями поиска)

Неделя 10. LCA

10.133. Дано дерево. Заранее известны q запросов вида: «прибавить ко всем ребрам на пути от u до v число x». Выведите вес каждого ребра после выполнения всех запросов. Подсказка: вспомните аналогичную задачу на массиве и используйте ту же технику.

10.134. Дано дерево, которое иногда меняется. Имеются две операции:

(a) Добавить новую вершину u и подвесить ее к вершине v

(b) Удалить лист дерева с номером u

Покажите, что можно пересчитывать двоичные подъемы без потери производительности по времени.

10.135. Дано дерево, на каждом ребре написано число. Требуется отвечать на запросы: «найти сумму чисел на пути от u до v» за $\mathcal{O}(\log n)$. Числа не меняются.

10.136. Дано дерево, на каждом ребре написано число. Требуется отвечать на запросы: «найти минимум чисел на пути от u до v» за $\mathcal{O}(\log n)$. Числа не меняются.

10.137. Дано дерево, у каждого ребра есть положительная длина. Требуется отвечать на запросы: «найти ближайшего предка v, длина пути до которого не меньше d» за $\mathcal{O}(\log n)$. Длины не меняются.

10.138. Дано дерево, у каждого ребра есть длина. Требуется отвечать на запросы: «найти середину пути от u до v» за $\mathcal{O}(\log n)$. Длины не меняются. Середина пути — это либо вершина, либо точка на ребре. Во втором случае достаточно найти ребро, на котором находится середина.

10.139. Дано дерево. Требуется отвечать на запросы: «найти длину пересечения двух путей» за $\mathcal{O}(\log n)$. Пути задаются парами вершин (u_1, v_1) и (u_2, v_2).

10.140. Дано подвешенное дерево. Требуется отвечать на запросы двух типов за $\mathcal{O}(\log n)$:

(a) Найти LCA вершин u и v

(b) Взять целиком поддерево вершины u и подвесить его к вершине v

10.141. Дано подвешенное дерево. Требуется отвечать на запросы двух типов за $\mathcal{O}(\log n)$:

(a) Найти LCA вершин u и v

(b) Переподвесить дерево за вершину v

10.142. Дано подвешенное дерево. Требуется отвечать на запросы: «дано множество из k вершин, найти их LCA» за $\mathcal{O}(k \log n)$.

10.143. Дано подвешенное дерево. Требуется отвечать на запросы: «дано множество из k вершин, найти их LCA» за $\mathcal{O}(k + \log n)$.

10.144. Дано дерево, у каждого ребра есть длина. Требуется отвечать на запросы двух типов за $\mathcal{O}(\log n)$ (препроцессинг $\mathcal{O}(n)$):

(a) Изменить длину ребра (u, v)

(b) Найти расстояние между вершинами u и v

10.145. Дано дерево. Требуется отвечать на запросы: «даны две вершины u и v, найдите количество вершин в дереве, равноудаленных от вершин u и v» за $\mathcal{O}(\log n)$.

10.146. Дано подвешенное дерево. Требуется отвечать на запросы: «является ли вершина u предком вершины v» за $\mathcal{O}(1)$. Препроцессинг: $\mathcal{O}(n)$.

10.147. Придумайте как используя предыдущую задачу сделать алгоритм поиска LCA через бинподъемы поднимая только одну вершину

Неделя 11. HLD

11.148. Какое максимальное и минимальное количество путей может получиться в Heavy-Light декомпозиции дерева?

11.149. Пусть в дереве веса на вершинах. Покажите, как можно решать задачи при помощи HLD в таком случае.

11.150. Пусть веса в дереве есть не на вершинах, а на ребрах. Покажите, как можно решать задачи при помощи HLD в таком случае.

11.151. Научитесь обрабатывать следующие запросы:

(a) Изменить вес ребра

(b) Найти самый далекий от корня лист

11.152. Дано дерево. Требуется для каждой вершины вычислить сумму расстояний от данной вершины до всех остальных. Время работы: $\mathcal{O}(n)$.

11.153. Дерево поджигают в некоторой вершине. Огонь распространяется по ребру за одну единицу времени. Вычислите для каждой вершины, за сколько времени сгорит все дерево, если поджечь его в данной вершине.

11.154. Научитесь отвечать на следующие запросы: «дерево подожгли в вершинах u и v, найти время, за которое все дерево сгорит».

11.155. Покажите, как при помощи HLD вычислять значение не коммутативной функции на пути.

11.156. Дано ориентированное дерево на n вершинах. Необходимо отвечать на запросы: «можно ли добраться из вершины u в вершину v, учитывая ориентацию ребер» за $\mathcal{O}(\log n)$.

11.157. Решите предыдущую задачу, добавив запросы: изменить ориентацию ребер на пути из u в v на противоположную.

11.158. Независимым множеством называется множество вершин, таких что никакие две вершины множества не соединены ребром. Найдите максимальное по размеру независимое множество в дереве за $\mathcal{O}(n)$.

11.159. Паросочетанием называется множество ребер, такое что никакие два ребра из множества не имеют общей вершины. Найдите максимальное по размеру паросочетание в дереве за $\mathcal{O}(n)$.

11.160. Дано дерево из n вершин. В вершине с номером 1 располагается ваш дом, а в листьях дерева находятся рестораны. В некоторых вершинах дерева сидят коты (вам известно, в каких именно вершинах). Вы хотите посетить ресторан, и для этого можете перемещаться по ребрам дерева. Однако, вы можете выбирать только такие маршруты, чтобы не посещать **подряд** m вершин, в которых находятся коты, так как у вас аллергия. Найдите количество ресторанов, до которых вы сможете добраться за $\mathcal{O}(n)$.

11.161. Дано дерево на n вершинах, у каждого ребра есть длина. Диаметром называется наидлиннейший простой путь в дереве. Научитесь искать диаметр в дереве за $\mathcal{O}(n)$.

11.162. Дано дерево на n вершинах. Изначально все вершиины не помечены. Требуется отвечать на запросы за $\mathcal{O}(\log n)$ в Online:

(a) Пометить вершину v

(b) Найти самого глубокого не помеченного общего предка вершин u и v

Препроцессинг: $\mathcal{O}(n \log n)$.