АиСД у2023. Второй семестр

Домашние задания М3132-М3133

⟨Версия от 26 мая 2024 г.⟩

Темы

Неделя 1. Дерево отрезков

В заданиях с 1.1 по 1.4 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 значение суммы $a_l - a_{l+1} + a_{l+2} - \ldots + (-1)^{r-l} a_{r-1}$.

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

1.3. 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.

1.4. 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.

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

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

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

1.8. Решите задачу 1.7, при условии, что элементы массива — произвольные целые числа.

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

1.10. 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.

1.11. 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.

1.12. 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: $\mathcal{O}((n+q)\log n)$.

1.13. 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: $\mathcal{O}((n+q)\log n)$.

1.14. Нужно реализовать две операции:

(a) Определить значение на позиции pos.

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

1.15. 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 $a_j - a_i \leq 1$. 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.

1.16. Придумайте как сделать реализацию функции `get` "снизу"для не коммутативной функции

Неделя 2. Дерево отрезков — 2

В заданиях с 17 по 23 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()`.

2.17. (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

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

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

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

2.22. (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

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

В заданиях с 2.24 по 2.27 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. Дерево Фенвика. Sparse Table

3.32. 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.

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

3.34. 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.

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

3.36. 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.

3.37. 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 $[l, r)$.

3.38. 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?

3.39. 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.

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

3.41. 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!

3.42. 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.

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

3.44. 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.

3.45. 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: $\mathcal{O}(n \log^3 n)$.

3.46. 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: $\mathcal{O}(k^2 + n \log n \log^2 k)$.

3.47. 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).

3.48. 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

3.49. 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.

(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 4. Двоичное дерево поиска. Декартово дерево

4.50. Напишите рекурсивную функцию, выводящую все элементы двоичного дерева поиска в порядке сортировки, за $\mathcal{O}(n)$.

4.51. Напишите 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.

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

4.53. 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.

4.54. 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.

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

4.56. 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 $\log n$.

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

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

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

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

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

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

4.63. 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 $[l, r]$

4.64. 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 $[l, r]$

4.65. 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 $\Omega(n)$.

4.66. 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.

4.67. 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 $\mathcal{O}(n)$.

4.68. Покажите, как на базе дерева поиска по неявному ключу сделать СНМ со временем работы $\mathcal{O}(\log n)$.

4.69. Покажите, как на базе дерева поиска реализовать мультимножесво, поддерживающее за время $\mathcal{O}(\log n)$ операции добавления и удаления элемента, а также запрос количества вхождений элемента.

Неделя 5. LCA

5.70. 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.

5.71. 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.

5.72. 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.

5.73. 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.

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

5.75. 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.

5.76. 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).

5.77. 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

5.78. 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

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

5.80. 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

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

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

Неделя 6. AVL и Splay

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

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

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

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

6.87. 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 $l \le x \le r$

6.88. 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.

6.89. 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).

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

6.91. Упростим операцию `splay`: будем всегда делать операцию `зиг`, забыв про существование `зиг-зиг` и `зиг-заг`. Покажите, что существует последовательность из n операций, которая в этом случае будет работать дольше, чем за $n \log n$.

6.92. Упростим операцию `splay`: будем вместо операции `зиг-заг` делать операцию `зиг-зиг`. Покажите, что существует последовательность из n операций, которая в этом случае будет работать дольше, чем за $n \log n$.

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

6.94. 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?

6.95. 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?

6.96. 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.

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

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

Неделя 7. HLD

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

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

7.101. Научитесь обрабатывать следующие запросы:

(a) Изменить вес ребра

(b) Найти самый далекий от корня лист

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

7.103. 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.

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

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

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

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

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

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

7.110. 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 $\mathcal{O}(n)$.

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

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

Неделя 8. Хеш-таблицы

8.113. Будем разрешать коллизии в хеш-таблице с помощью списков, но хранить списки в отсортированном по ключам порядке. Как это повлияет на время работы в худшем случае и в среднем?

8.114. Добавьте в хеш-таблицу возможность проитерироваться по всем ее элементам в том порядке, в котором они добавлялись за $\mathcal{O}(n)$.

8.115. Множество хеш-функций называется 2-универсальным, если для любой пары ключей (x, y) $(x \neq y)$ пара $(h(x), h(y))$ может принять любую из возможных m^2 пар значений равновероятно (будем считать, что число m — простое). Докажите, что множество хеш-функций $h_a(x) = ax \pmod m$ не является 2-универсальным.

8.116. Докажите, что множество хеш-функций $h_{ab}(x) = ax + b \pmod m$ является 2-универсальным.

8.117. Научитесь удалять ключи из хеш-таблицы с открытой адресацией за $\mathcal{O}(1)$ в среднем.

8.118. Найдите реализацию метода `Long.hashCode()` в языке Java. Как быстро сгенерировать много объектов типа `Long` с одинаковым хешом?

8.119. Найдите реализацию метода `String.hashCode()` в языке Java. Как быстро сгенерировать много объектов типа `String` с одинаковым хешом?

8.120. Придумайте, как реализовать операцию `merge` для объединения двух множеств, хранящихся в виде хеш-таблиц. Амортизированная стоимость данной операции должна быть $\mathcal{O}(\log n)$.

8.121. Какой размер множества одинаковых равномерно распределенных от 1 до n случайных величин необходим, чтобы вероятность того, что хотя бы две из них совпадут, была хотя бы $\frac{1}{2}$? Сделайте вывод о вероятности коллизии в хеш-таблице, если при ее реализации мы игнорируем коллизии.

Неделя 9. Корневые оптимизации

9.122. Пусть дана задача о рюкзаке: даны n предметов с весами $w_1, w_2, \ldots, w_n$. Требуется определить, можно ли набрать суммарный вес, в точности равный W. Также известно, что $\sum w_i = S$. Мы уже умеем решать задачу за $\mathcal{O}(n \cdot W)$. Научитесь решать задачу за $\mathcal{O}(S\sqrt{S})$.

9.123. Дано изначально пустое множество строк D. Требуется отвечать на запросы:

- ▷ Добавить строку s в множество D.
- ▷ Удалить строку s из множества D.
- ▷ Для заданной строки s найти количество вхождений строк из множества D в нее. Если некоторая строка p из множества D имеет несколько вхождений в s, необходимо посчитать их все.

Сумма длин строк во всех запросах равна L. Требуется решить задачу за $\mathcal{O}(L\sqrt{L})$.

9.124. Дан неориентированный граф на n вершинах и m ребрах. В каждой вершине записано целое число. Необходимо отвечать на запросы двух типов:

- ▷ Прибавить ко всем соседям вершины v число x
- ▷ Вывести значение, хранящееся в вершине v

Асимптотика: $\mathcal{O}((m+q)\sqrt{m})$.

9.125. Дано дерево на n вершинах. Изначально все вершины не покрашены. Необходимо отвечать на запросы двух типов:

- ▷ Покрасить вершину v.
- ▷ Найти ближайшую к v покрашенную вершину.

Асимптотика: $\mathcal{O}((n+q)\sqrt{n+q})$. Можно сделать препроцессинг.

9.126. Дан массив из n целых чисел. Есть m запросов к массиву, каждый запрос описывается парой целых чисел l_j и r_j $(1 \le l_j \le r_j \le n)$. Для каждого запроса l_j, r_j необходимо посчитать количество таких чисел x, что число x встречается ровно x раз среди чисел $a_{l_j}, a_{l_{j+1}}, \ldots, a_{r_j}$.

9.127. Имеется массив натуральных чисел $a_1, a2, \ldots, a_n$. Рассмотрим некоторый его подмассив $a_l, a_{l+1}, \ldots, a_r$, где $1 \le l \le r \le n$, и для каждого натурального числа s обозначим через K_s число вхождений числа s в этот подмассив. Назовем мощностью подмассива сумму произведений $K_s \cdot K_s \cdot s$ по всем различным натуральным s.

9.128. Есть N лунок, расположенных в ряд, пронумерованных слева направо числами от 1 до N. У каждой лунки изначально установлена своя сила выброса (у лунки с номером i она равна a_i). Если вбросить шарик в лунку i, то он тут же вылетит из нее и попадет в лунку $i + a_i$, после чего он опять вылетит и т.д.. Если же лунки с таким номером нету, то он просто вылетит за край ряда. Есть M запросов:

- ▷ Установить силу выброса лунки a равной b.
- ▷ Вбросить шарик в лунку a и посчитать количество прыжков шарика, прежде чем он вылетит за край ряда, а так же записать номер лунки, после выпрыгивания из которой шарик вылетел за край.

Нужно научиться отвечать на запросы второго типа

9.129. Малыш достал из кладовки N банок варенья и выставил их в ряд. В банке номер i содержится ровно a_i грамм варенья. Карлсон немного подумал и решил, что в некоторых банках недостаточно варенья, и что в банке номер i должно быть хотя бы b_i грамм варенья.

Выходить из этой ситуации Карлсон хочет в M этапов. На каждом этапе он выбирает числа l, r, x, y, а затем выполняет следующие операции: в банку номер l он добавляет x грамм варенья, в банку номер $l + 1$ — $x + y$ грамм варенья, в банку номер $l + 2$ — $x + y \cdot 2$, и так далее. В банку номер r наш герой добавит $x + y \cdot (r - l)$ грамм варенья.

Малышу хочется определить для каждой банки i наименьший номер операции, после которой в ней станет хотя бы b_i грамм варенья. Помогите Малышу: найдите соответствующее число для каждой банки.

9.130. Дан массив из n целых чисел и q запросов одного из трех типов:

- ▷ Найти минимум на отрезке $[l, r]$.
- ▷ Развернуть подотрезок массива $[l, r]$.
- ▷ Прибавить ко всем элементам отрезка $[l, r]$ число x.

Время работы: $\mathcal{O}((n + q)\sqrt{n + q})$.

Неделя 10. Вычислительная геометрия — 1

10.131. При помощи предиката поворота проверьте, что точка P лежит внутри (или на границе) треугольника ABC.

10.132. Проверьте, что точка P лежит внутри выпуклого многоугольника. Время работы: $\mathcal{O}(n)$.

10.133. Даны точки A, O, B и P. При помощи предиката поворота проверьте, что точка P лежит внутри угла AOB.

10.134. Иван заблудился в саванне, и на его поиски выдвинулись n львов. К сожалению, львы ищут Ивана не для того, чтобы помочь ему, а для того чтобы поужинать. В начальный момент времени Иван находится в точке P_0. Он выбирает направление, в котором он будет ехать, спасаясь от львов, и дальше все время едет по прямой. Львы, будучи великолепными охотниками, сразу определяют направление, выбранное Иваном, и планируют его поимку соответственно. В частности, львы понимают, что Иван поедет по прямой. Скорости всех львов постоянны и равны между собой, а также равны скорости передвижения Ивана. Изначально львы находятся в точках $P_1, P_2, \ldots, P_n$. Определите, сможет ли Иван спастись от львов, или ему неминуемо придется стать ужином. В случае, если спастись удастся, определите направление, в котором нужно двигаться Ивану. Считается, что Иван станет ужином, если в какой-то момент времени координаты Ивана и хотя бы одного льва совпадут. Время работы: $\mathcal{O}(n \log n)$.

10.135. Дан выпуклый многоугольник, состоящий из n вершин. Поступают запросы, каждый из которых характеризуется прямой $a_i x + b_i y + c_i = 0$. Научитесь проверять, пересекается ли данная прямая с многоугольником. Время работы на запрос: $\mathcal{O}(\log n)$.

10.136. Даны n точек $P_1, P_2, \ldots, P_n$. Поступают запросы, каждый из которых характеризуется прямой $a_i x + b_i y + c_i = 0$. Научитесь находить точку P_i, расстояние от которой до прямой максимально. Время работы на запрос: $\mathcal{O}(\log n)$.

10.137. Даны два выпуклых многоугольника с n и m вершинами, соответственно. Проверьте, что многоугольники пересекаются за $\mathcal{O}(nm)$.

10.138. Найдите расстояние между двумя выпуклыми многоугольниками за $\mathcal{O}(nm)$.

10.139. Нужно выйти из точки (x_1, y_1), пройти к реке, набрав там воды, принести ее в точку (x_2, y_2). задающие уравнение реки $Ax + By + C = 0$, необходимо найти кратчайший путь.

10.140. Вам заданы две окружности. Необходимо выяснить, пересекаются ли заданные окружности и найти точки их пересечения.

10.141. Дан набору из n прямых, определяемых уравнениями $y = k_i \cdot x + b_i$. Требовалось определить, существует ли хотя бы одна точка пересечения двух прямых, лежащая строго внутри полосы ограниченной $x_1 < x_2$. Другими словами, правда ли что найдутся такие $1 \le i < j \le n$ и такие x', y', что:

- $y' = k_i \cdot x' + b_i$, то есть точка (x', y') принадлежит прямой с номером i;
- $y' = k_j \cdot x' + b_j$, то есть точка (x', y') принадлежит прямой с номером j;

- ▷ $x_1 < x' < x2$, то есть точка (x', y') лежит внутри полосы ограниченной $x_1 < x_2$.

$(2 \leq n \leq 100000)$ — количество прямых в задании

Неделя 11. Вычислительная геометрия — 2

11.142. Как меняется сумма Минковского двух фигур при параллельном переносе одной из них на некоторый вектор? А при замене системы координат?

11.143. Докажите, что сумма Минковского двух выпуклых многоугольников — это выпуклый многоугольник.

11.144. Найдите расстояние между двумя выпуклыми многоугольниками за $\mathcal{O}(n+m)$.

11.145. Дан выпуклый многоугольник. Требуется отвечать на запросы: «дана точка P, необходимо найти касательные к многоугольнику, проходящие через данную точку». Время работы: $\mathcal{O}(\log n)$ на запрос.

11.146. Дан выпуклый многоугольник. Найдите длину максимального отрезка, концы которого лежат на границе многоугольника. Время работы: $\mathcal{O}(n)$.

11.147. Дан произвольный многоугольник. Постройте его триангуляцию за $\mathcal{O}(n^2)$. Триангуляцией называется разбиение многоугольника на $n-2$ треугольника, вершинами которых являются вершины исходного многоугольника, при условии, что все треугольники лежат внутри заданного многоугольника.

11.148. Докажите, что для любых трех хороших (связных, непрерывных, замкнутых, бла-бла-бла), но не обязательно выпуклых, множеств S_1, S_2 и S_3 верно следующее свойство: $S_1 + (S_2 \cup S_3) = (S_1 + S_2) \cup (S_1 + S_3)$, где операция $+$ — это сумма Минковского.

11.149. Как выглядит сумма Минковского двух невыпуклых многоугольников?

11.150. Даны m выпуклых многоугольников с размерами $n_1, n_2, \ldots, n_m$. Пусть $n = n_1 + n_2 + \ldots + n_m$. Найдите сумму Минковского всех многоугольников за $\mathcal{O}(n \log n)$.

11.151. Дан выпуклый многоугольник на n вершинах — это стена. Снаружи от него даны m точек — в них расположены фонари. Фонарь светит во все стороны на любое расстояние. Определить длину освещенной части стены. Время работы: $\mathcal{O}(n \log n)$.

11.152. Даны два множества точек. Найдите прямую, которая их разделяет, то есть такую прямую, чтобы все точки одного множества были по одну сторону, а все точки второго — по другую. Время работы: $\mathcal{O}(n \log n)$.

Неделя 12. Центроидная декомпозиция

12.153. Дано дерево, в котором у любой вершины степень не превосходит 3. Докажите, что существует ребро, после удаления которого дерево распадется на компоненты, размер которых не превосходит $\frac{2}{3}n + 1$.

12.154. Докажите, что в дереве всегда есть центроид.

12.155. Докажите, что в дереве не более двух центроидов.

12.156. Задано подвешенное дерево. Найдите для каждого поддерева его центроид. Время $\mathcal{O}(n)$.

12.157. Дано дерево, каждое ребро имеет длину. Научитесь отвечать на запросы вида: «найти количество вершин, находящихся на расстоянии, не превосходящим d, от вершины v», если есть запросы на изменение длины ребра.

12.158. Дано дерево, у каждой вершины есть положительный вес. Научитесь отвечать на запросы двух типов:

(a) Изменить вес вершины

(b) Найти путь максимального веса, проходящий через вершины v

12.159. Дано дерево, вершины покрашены в черный и белый цвета. Научитесь отвечать на запросы:

(a) Изменить цвет вершины

(b) Найти сумму расстояний от v до всех вершин того же цвета

12.160. Дано дерево, вершины покрашены в черный и белый цвета. Научитесь отвечать на запросы:

(a) Изменить цвет вершины

(b) Найти ближайшую к v вершину того же цвета

12.161. Дано дерево, в каждой вершине записано число c_v. Научитесь отвечать на запросы:

(a) Изменить число, записанное в вершине

(b) Вычислить значение $\sum\limits_u \sum\limits_v c_{\mathrm{lca}(u,v)}$

12.162. Дана дорожная сеть в виде дерева, которую вечно ремонтируют. Ремонт производится этапами. Этап (v, d) состоит в том, что ремонтируются все ребра на расстоянии, не превосходящем d, от вершины v. Научитесь отвечать на запросы:

(a) Провести этап ремонта с параметрами (v, d)

(b) Узнать, когда последний раз ремонтировалось ребро

12.163. Дано дерево, над которым производят эксперименты. Каждый эксперимент задается парой чисел (v, d). В рамках эксперимента в вершине v разливают токсичное вещество с уровнем вони d. После этого во всех вершинах u, находящихся на расстоянии, не превосходящем d, от вершины v, уровень вони увеличивается на $d - \text{dist}(u, v)$. Научитесь отвечать на запросы:

(a) Провести эксперимент с параметрами (v, d)

(b) Вычислить уровень вони в вершине v

12.164. Дано дерево, у каждой вершины есть цвет. Изначально все вершины покрашены в цвет 0. Научитесь отвечать на запросы:

(a) Покрасить все вершины, расположенные на расстоянии, не превосходящем d, от вершины v, в цвет c

(b) Найти цвет вершины v

12.165. Два дерева T_1 и T_2 называются изоморфными, если существует такая биекция f из множества вершин первого дерева в множество вершин второго дерева, что в дереве T_1 существует ребро (u, v) тогда и только тогда, когда в дереве T_2 существует ребро $(f(u), f(v))$. Два подвешенных дерева называются изоморфными, если f также переводит корень первого дерева в корень второго дерева. Научитесь проверять подвешенные деревья на изоморфизм за $\mathcal{O}(n \log n)$.

12.166. Научитесь проверять неподвешенные деревья на изоморфизм за $\mathcal{O}(n \log n)$.