

## 1 Дерево отрезков

Каждая подзадача сдается отдельно. Все операции должны работать за  $O(\log n)$ , если иное не указано в задании.

- 1.1. Для каждой задачи из этого блока сформулируйте, для какого множества значений и какой ассоциативной операции нужно построить дерево отрезков. Структуру самого дерева и алгоритм выполнения операций изменять нельзя.

Есть массив  $a$  из  $n$  целых чисел и две операции: присвоить значение:  $a_i = x$ , и одна из следующих:

- (1.1.1) найти минимум на отрезке от  $l$  до  $r$ , а так же число элементов, равных этому минимуму
  - (1.1.2) найти минимум на отрезке от  $l$  до  $r$ , а так же индекс самого левого элемента, равного этому минимуму
  - (1.1.3) найти значение суммы  $a_l - a_{l+1} + a_{l+2} - a_{l+3} + \dots \pm a_{r-1}$
  - (1.1.4) найти значение суммы  $a_l + 2a_{l+1} + 3a_{l+2} + \dots + (r-l)a_{r-1}$
  - (1.1.5) найти на заданном отрезке  $[l, r]$  подотрезок  $[l_1, r_1]$  ( $l \leq l_1 \leq r_1 \leq r$ ), сумма на котором максимальна (достаточно вывести эту сумму, но можно и отрезок тоже)
- 1.2. В задачах из этого блока нужно добавить новую операцию в дерево отрезков.
- Есть массив  $a$  из  $n$  целых чисел и две операции: присвоить значение:  $a_i = x$ , и одна из следующих:
- (1.2.1) найти минимальное  $i$  для которого  $a_i \geq k$
  - (1.2.2) вывести все  $i$  для которых  $a_i \geq k$  за время  $O(x \log n)$ , где  $x$  — размер ответа
  - (1.2.3) найти минимальное  $i$  на отрезке от  $l$  до  $r$ , для которого  $a_i \geq k$
- 1.3. Есть парковка на  $n$  мест. Каждое место может быть занятым или свободным. Нужно обрабатывать операции: пометить место как занятое/свободное, и одна из следующих:
- (1.3.1) найти число свободных мест на отрезке от  $l$  до  $r$
  - (1.3.2) найти  $k$ -е по порядку свободное место
  - (1.3.3) найти свободное место, ближайше к  $i$
- 1.4. Есть строка из  $n$  круглых скобок. Нужно обрабатывать запросы: 1) изменить  $i$ -ю скобку, 2):
- (1.4.1) проверить, является ли правильной скобочной последовательностью подстрока с  $l$  до  $r$
  - (1.4.2) найти наибольший префикс подстроки с  $l$  до  $r$ , который является правильной последовательностью

## 2 Дерево отрезков-2

- 2.1. Есть массив  $a$  из  $n$  булеанов. Нужно обрабатывать запросы за  $O(\log n)$ :

- (2.1.1) присвоить значение  $x$  всем элементам отрезка, найти ближайшую к  $i$  единицу
- (2.1.2) изменить значение всех элементов отрезка на противоположное, найти число единиц на отрезке
- (2.1.3) присвоить значение  $x$  всем элементам отрезка, выполнить `for (i = 1 .. r - 1) : a[i] = a[i] and a[i - 1]`, выполнить `for (i = 1 .. r - 1) : a[i] = a[i] or a[i - 1]`, найти число единиц на отрезке

- (2.1.4) присвоить значение  $x$  всем элементам отрезка,  
найти число непрерывных отрезков из единиц
  - (2.1.5) присвоить значение  $x$  всем элементам отрезка,  
найти самый длинный непрерывный отрезок из единиц
- 2.2. Есть массив  $a$  из  $n$  целых чисел. Нужно обрабатывать запросы за  $O(\log n)$ :
- (2.2.1) присвоить значение  $x$  всем элементам отрезка,  
изменить элементы отрезка  $a_i = -a_i$ ,  
найти сумму на отрезке
  - (2.2.2) присвоить значение  $x$  всем элементам отрезка,  
изменить элементы отрезка  $a_i = -a_i$ ,  
найти максимум на отрезке
  - (2.2.3) присвоить значение  $x$  всем элементам отрезка,  
изменить элементы отрезка  $a_i = -a_i$ ,  
найти подотрезок с максимальной суммой
  - (2.2.4) присвоить значение  $x$  всем элементам отрезка,  
прибавить  $x$  ко всем элементам отрезка,  
найти значение элемента
  - (2.2.5) присвоить значение  $x$  всем элементам отрезка,  
прибавить  $x$  ко всем элементам отрезка,  
найти сумму на отрезке
  - (2.2.6) изменить элементы отрезка  $a_i = \max(a_i, x)$ ,  
найти минимум на отрезке
  - (2.2.7) изменить элементы отрезка  $a_i = \max(a_i, x)$ ,  
найти максимум на отрезке
  - (2.2.8) изменить элементы отрезка  $a_i = \max(a_i, x)$ ,  
изменить элементы отрезка  $a_i = \min(a_i, x)$ ,  
найти значение  $a_i$ .
  - (2.2.9) присвоить значение  $x$  всем элементам отрезка,  
найти самый длинный отрезок из одинаковых чисел
  - (2.2.10) изменить элементы отрезка  $a_i = a_i + x \cdot i + y$ ,  
найти сумму на отрезке
  - (2.2.11) присвоить элементам отрезка значения  $a_i = x \cdot i + y$ ,  
найти максимум на отрезке
  - (2.2.12) присвоить элементам отрезка значения  $a_i = x \cdot i + y$ ,  
найти НОД чисел на отрезке
  - (2.2.13) присвоить элементам отрезка значения  $a_i = x \cdot i + y$ ,  
найти отрезок с максимальной суммой

### 3 Еще задачи на дерево отрезков и смежные темы

- 3.1. Постройте дерево Фенвика по заданному массиву за  $O(n)$ .
- 3.2. Постройте дерево Фенвика по заданному массиву без дополнительной памяти за  $O(n)$ .
- 3.3. Дано дерево Фенвика, восстановите исходный массив без дополнительной памяти за  $O(n)$ .
- 3.4. Как изменятся время работы и объем используемой памяти разреженной таблицы, если хранить отрезки длиной не  $2^k$ , а  $x^k$  ( $x > 2$ )?

- 3.5. Добавить в дерево Фенвика операцию, которая находит максимальный префикс массива, на котором сумма не больше  $x$  (все значения неотрицательные) за  $O(\log n)$ .
- 3.6. Есть шахматное поле  $n \times n$ . Обрабатывать запросы: 1) добавить/удалить ладью, 2) найти число клеток в заданном прямоугольнике, которые не бьются ни одной ладьей. Оба запроса за  $O(\log n)$ .
- 3.7. Последовательность  $f_i$  вычисляется по следующим правилам:  $f_{-1} = f_0 = 1$ ,  $f_i = (a_i \cdot f_{i-1} + b_i \cdot f_{i-2}) \pmod{M}$ . Нужно обрабатывать запросы: 1) для заданного  $i$  изменить числа  $a_i$  и  $b_i$  на  $x$  и  $y$  (и пересчитать последовательность), 2) найти значение  $f_i$ . Оба запроса за  $O(\log n)$ .
- 3.8. Есть полоса из  $n$  клеток, в некоторых могут находиться препятствия. Есть кузнечик, который может прыгать на одну, две или три клетки. Обрабатывать запросы: 1) добавить/удалить препятствие, 2) найти число способов добраться из клетки  $x$  в клетку  $y$ . Оба запроса за  $O(\log n)$ .
- 3.9. Есть два массива  $a$  и  $b$ . Нужно обрабатывать запросы: 1) скопировать участок массива  $a$  в массив  $b$  (то есть, сделать  $b_{y+q} = a_{x+q}$  для всех  $q$  от 0 до  $k-1$ ) 2) найти значение  $b_i$ . Оба запроса за  $O(\log n)$ .
- 3.10. Есть город из  $n$  домов, выстроенных в ряд, высота  $i$ -го дома равна  $a_i$ . На город последовательно падает  $m$  бомб. Бомба с силой  $p_j$ , попавшая в дом  $x_j$ , разрушает все еще не разрушенные дома  $i$ , для которых  $a_i \leq p_j - |x_j - i|$ . Найдите для каждого дома, какая по счету бомба его разрушит за  $O((n+m) \log n)$ .
- 3.11. Про некоторый массив  $a$  из  $n$  чисел известно  $m$  свойств вида  $\max(a_l..a_r) = x$ . Постройте массив, удовлетворяющий всем свойствам или скажите, что это невозможно.
- 3.12. Про некоторый массив  $a$  из  $n$  чисел известно  $m$  свойств вида  $\sum(a_l..a_r) = x$ . Постройте массив, удовлетворяющий всем свойствам или скажите, что это невозможно.
- 3.13. В этой задаче можно использовать только обычное дерево Фенвика. Есть массив  $a$  из  $n$  чисел. Нужно обрабатывать запросы: 1) прибавить  $x$  ко всем элементам отрезка, 2) найти значение  $a_i$ .
- 3.14. В этой задаче можно использовать несколько деревьев Фенвика. Есть массив  $a$  из  $n$  чисел. Нужно обрабатывать запросы: 1) прибавить  $x$  ко всем элементам отрезка, 2) найти сумму  $a_i$  на отрезке.
- 3.15. Есть подвешенное дерево из  $n$  узлов (не обязательно двоичное). В каждом узле написано число. Обрабатывать запросы 1) задать значение в узле 2) найти максимум в поддереве.
- 3.16. Есть массив  $a$  (он не меняется). Отвечать на запросы: найти сумму элементов на отрезке  $a[l..r]$ , значения которые лежат в интервале от  $x$  до  $y$ .
- 3.17. Есть массив  $a$  (он не меняется). Отвечать на запросы: есть ли элемент, который встречается на отрезке  $a[l..r]$  больше раз, чем все остальные в сумме (то есть занимает больше половины отрезка)?

## 4 Двумерные структуры

- 4.1. Есть несколько непересекающихся прямоугольников. Можно за один ход перепрыгнуть с одного прямоугольника на другой, если есть вертикальный или горизонтальный отрезок, соединяющий эти прямоугольники и не пересекающий другие прямоугольники. Найдите для каждого прямоугольника список прямоугольников, на которые с него можно прыгнуть  $O(n \log n)$ .

- 4.2. Есть несколько прямоугольников. Они расположены так, что их границы не пересекаются, но один прямоугольник может быть внутри другого. Постройте дерево вложенности прямоугольников за  $O(n \log n)$ .
- 4.3. Таблица  $n \times n$ , обрабатывать запросы: 1) прибавить значение к прямоугольнику, 2) найти значение в ячейке. Оба запроса за  $O(\log^2 n)$ .
- 4.4. Таблица  $n \times n$ , обрабатывать запросы: 1) прибавить значение к прямоугольнику, 2) найти сумму значений в прямоугольнике. Оба запроса за  $O(\log^2 n)$ . (Решите с помощью нескольких двумерных деревьев Фенвика.)
- 4.5. Таблица  $n \times n$ . После предподсчета за  $O(n^2)$  отвечать на запросы: по данным  $(x, y, d)$  находить сумму в прямоугольном треугольнике с вершинами в  $(x, y)$ ,  $(x + d, y)$  и  $(x, y + d)$ .
- 4.6. Та же задача, но есть еще операция изменить элемент таблицы.
- 4.7. Представьте, что вам нужно построить дерево отрезков размера  $M$  и совершить с ним  $n$  операций. При этом  $n$  сильно меньше  $M$ . Попробуем сэкономить память, храня только интересные узлы.
- (4.7.1) Покажите, как получить структуру, в которой  $O(n \log M)$  узлов.
- (4.7.2) Покажите, как получить структуру, в которой  $O(n)$  узлов.
- 4.8. Примените идеи из предыдущей задачи к двумерному дереву. Какие результаты получатся?
- 4.9. 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.
- (4.9.1) За  $O(k^2 + n \log n \log^2 k)$ .
- (4.9.2) За  $O(n(\log n + \log^2 k))$ .
- 4.10. 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?  $O(n \log^3 n)$ .
- 4.11. Манхеттенское расстояние на плоскости задается как сумма расстояний по координатам  $|x_1 - x_2| + |y_1 - y_2|$ . Дана таблица  $n \times n$ , в некоторых клетках которой находятся фишки. Нужно по запросу  $(x, y, r)$  за  $O(\log^2 n)$  находить число фишек, находящихся на манхеттенском расстоянии не больше  $r$  от точки  $(x, y)$ .

## 5 Деревья поиска

- 5.1. Напишите рекурсивную процедуру, выводящую элементы дерева поиска в отсортированном порядке, за время  $O(n)$ .
- 5.2. Напишите нерекурсивную процедуру, выводящую элементы дерева поиска в отсортированном порядке, за время  $O(n)$  с  $O(1)$  дополнительной памяти (у узлов есть указатели на родителей).
- 5.3. Докажите, что нет алгоритма, который строит дерево поиска по заданному массиву из  $n$  элементов быстрее, чем за  $O(n \log n)$  в худшем случае.
- 5.4. Для каждого узла  $x$  посчитайте число  $w(x)$ , равное числу узлов в его поддереве (включая сам  $x$ ). Время  $O(n)$ .

- 5.5. Используя вычисленные значения  $w(x)$ , научитесь находить  $k$ -й по возрастанию элемент дерева. Время  $O(H)$ .
- 5.6. Используя  $w(x)$ , научитесь находить по заданному ключу  $x$  число элементов, меньших  $x$ . Время  $O(H)$  ( $H$  — высота дерева).
- 5.7. По данному узлу дерева найдите следующие  $k$  узлов в отсортированном порядке за время  $O(H + k)$ .
- 5.8. Приведите пример дерева, в котором средняя глубина узла (среднее расстояние от узла до корня)  $O(\log n)$ , а высота дерева (максимальное расстояние от узла до корня) —  $\omega(\log n)$  (асимптотически больше).
- 5.9. Приведите пример AVL-дерева, в котором при добавлении нужно будет совершить процедуру балансировки более чем в одном узле.
- 5.10. Приведите пример AVL-дерева, в котором при удалении нужно будет совершить процедуру балансировки более чем в одном узле.
- 5.11. Приведите пример двух AVL-деревьев, хранящих одно и то же множество элементов, но имеющих разную высоту.
- 5.12. Двоичное дерево поиска называется красно-черным, если каждая его вершина раскрашена либо в красный, либо в черный цвет, родитель любой красной вершины черный, и путь до любого листа содержит одно и то же количество черных вершин. Докажите, что высота такого дерева  $O(\log n)$ .
- 5.13. Двоичное дерево поиска называется сбалансированным по весу, если для любой вершины  $v$  выполняется  $w(v) \geq \lfloor \alpha \cdot w(\text{parent}(v)) \rfloor$ , где  $w(v)$  — число вершин в поддереве,  $\alpha$  — какая-то положительная константа. Докажите, что высота такого дерева  $O(\log n)$ .
- 5.14. Проверить, что заданное дерево является корректным деревом поиска. Время  $O(n)$ .
- 5.15. Пусть в дереве для каждого узла высота его детей отличается не более чем на 5. Правда ли, что высота такого дерева  $O(\log n)$ ?
- 5.16. Пусть в дереве для каждого узла высота его детей отличается не более чем в два раза. Правда ли, что высота такого дерева  $O(\log n)$ ?

## 6 Деревья поиска - 2

- 6.1. Дан массив пар  $x, y$ , отсортированный по  $x$ . Постройте по нему декартово дерево за  $O(n)$ .
- 6.2. Дан массив пар  $x, y$ . Все  $x$  различны, а  $y$  могут совпадать. Проверить, что декартово дерево можно построить единственным образом.
- 6.3. Дан массив пар  $x, y$ . Все  $x$  различны, а  $y$  могут совпадать. Найти число способов построить декартово дерево.
- 6.4. Покажите, как на базе дерева поиска по неявному ключу сделать СНМ со временем работы  $O(\log n)$ .
- 6.5. Реализуйте в декартовом дереве по неявному ключу операцию  $\text{splitAfter}(x)$ , которая разделяет дерево, содержащее узел  $x$ , на деревья до  $x$ , включительно, и после  $x$ . Считайте, что размеры поддеревьев **не посчитаны**, зато у всех элементов есть ссылка на родителя в дереве.
- 6.6. Покажите, что если взять обычное дерево поиска без всяких балансировок и добавлять в него элементы в порядке убывания  $y$ , то получится декартово дерево.

- 6.7. Пусть у дерева поиска нет узлов с одним ребенком (то есть у каждой внутренней вершины ровно два ребенка). Правда ли, что высота такого дерева  $O(\log n)$ ?
- 6.8. Пусть в дереве размер (число вершин) для любого поддерева равен  $2^k - 1$  для какого-то целого  $k$ . Правда ли, что высота такого дерева  $O(\log n)$ ?
- 6.9. Дан массив чисел от 1 до  $n$ . Научитесь с помощью декартова дерева по неявному ключу обрабатывать запросы за  $O(\log n)$ : 1) развернуть отрезок массива от  $l$  до  $r$  задом наперед, 2) найти  $i$ -й элемент.
- 6.10. Дан массив чисел от 1 до  $n$ . Научитесь с помощью декартова дерева по неявному ключу обрабатывать запросы за  $O(\log n)$ : 1) для данных  $l$  и  $r$  поменять местами пары элементов:  $l$  и  $l + 1$ ,  $l + 2$  и  $l + 3$ , ...,  $r - 1$  и  $r$ , 2) найти  $i$ -й элемент.
- 6.11. Будем строить декартово дерево, не храня ключи  $y$ . В тех местах кода, где производится сравнение  $y$ , выберем случайный вариант с вероятностью 50%. Покажите, что после такого изменения некоторая последовательность действий может привести к тому, что высота дерева (вернее, ее матожидание) будет  $\Omega(n)$ .
- 6.12. 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.
- 6.13. Покажите, что структура дерева из предыдущего задания изоморфна обычному дереву поиска на  $n - 1$  элементах, а значит его можно балансировать всеми теми же способами.
- 6.14. Есть  $n$  окружностей. Они не пересекаются, но могут лежать внутри друг друга. Постройте дерево вложенности за  $O(n \log n)$ .
- 6.15. Есть несколько кругов. Они не пересекаются, но могут касаться. Постройте для каждого круга список тех, с кем он касается, за  $O(n \log n)$ .
- 6.16. Есть несколько многоугольников, в них суммарно  $n$  вершин. Они не пересекаются, но могут лежать внутри друг друга. Постройте дерево вложенности за  $O(n \log n)$ .

## 7 Splay деревья

- 7.1. Пусть вы сделали splay для вершины, которая находилась на глубине  $H$ . Посчитайте, как изменится сумма глубин вершин на этом пути после операции.
- 7.2. Покажите, что если делать splay просто каждый раз делая zig, то есть последовательность из  $n$  операций, которая работает дольше  $n \log n$ .
- 7.3. Пусть в splay дереве хранятся числа от 1 до  $n$ . Будем делать  $m$  операций: `splay(1)`, `splay(n)`, `splay(1)`, `splay(n)`, ... За какое суммарное время они будут работать?
- 7.4. Пусть в splay дереве размера  $n$  вы совершаете много действий над небольшим подмножеством из  $k$  элементов. Как будет выглядеть дерево? За какое время будут работать операции?
- 7.5. Пусть в AVL дереве хранятся числа от 1 до  $n$ . Покажите, что если делать операцию `find` по порядку для всех чисел от 1 до  $n$ , то суммарное время работы будет больше  $O(n)$ .
- 7.6. Пусть в splay дереве хранятся числа от 1 до  $n$ . Покажите, что если делать операцию `find` по порядку для всех чисел от 1 до  $n$ , то суммарное время работы будет  $O(n)$ .

- 7.7. Пусть в splay дереве хранятся числа от 1 до  $n$ . К дереву делают  $m$  запросов, суммарное число запросов к элементу  $i$  равно  $p_i$ . Покажите, что суммарное время запросов  $O(m + \sum p_i \log \frac{m}{p_i})$  (подсказка: нужно выбрать правильные  $w(v)$ , чтобы  $r(x)$  был таким, каким нужно).
- 7.8. Пусть в splay дереве хранятся числа от 1 до  $n$ . К дереву делают  $m$  запросов  $x_i$ . Покажите, что для любого элемента  $f$  суммарное время запросов  $O(m + n \log n + \sum \log(|x_i - f| + 1))$ .
- 7.9. Пусть в splay дереве хранятся числа от 1 до  $n$ . К дереву делают  $m$  запросов  $x_i$ . При этом известно, что  $|x_i - x_{i+1}| \leq D$ . Покажите, что суммарное время запросов  $O(m \log D)$ .
- 7.10. Покажите, как делать операцию **split** в splay дереве. За какое время она будет работать? (не забудьте посчитать изменение потенциала).
- 7.11. Покажите, как делать операцию **merge** в splay дереве. За какое время она будет работать? (не забудьте посчитать изменение потенциала).
- 7.12. Покажите, как построить splay дерево по заданному отсортированному массиву за  $O(n)$ . (не забудьте посчитать изменение потенциала).

## 8 Scapegoat Tree и другие задачи

- 8.1. Countdown Tree это дерево поиска, похожее на Scapegoat Tree. Оно устроено следующим образом. Когда создается узел  $v$ , в него записывается число, равное  $\alpha \cdot w(v)$ . Каждый раз, когда в поддереве узла совершается операция, это число уменьшается на 1. Когда число в узле становится равным 0, поддерево узла перестраивается с нуля.
- (8.1.1) Покажите, что высота такого дерева в каждый момент времени  $O(\log n)$ .
- (8.1.2) Покажите, амортизированное время операции добавления на таком дереве  $O(\log n)$ .
- 8.2. Dynamite Tree это еще одно дерево поиска, похожее на Scapegoat Tree. Оно устроено следующим образом. В каждом узле храним значение  $w(v)$ . Когда совершается операция в поддереве вершины  $v$ , мы бросаем монетку, и с вероятностью  $1/w(v)$  ломаем это поддерево и перестраиваем его с нуля.
- (8.2.1) Покажите, что матожидание высоты такого дерева в каждый момент времени  $O(\log n)$ .
- (8.2.2) Покажите, матожидание времени операции добавления на таком дереве  $O(\log n)$ .
- 8.3. Как сделать операцию **merge** в Scapegoat дереве за амортизированное  $O(\log n)$ ?
- 8.4. Как сделать операцию **split** в Scapegoat дереве за амортизированное  $O(\log n)$ ?
- 8.5. Было дерево поиска, его узлы выписали в порядке «корень, левое поддерево, правое поддерево». Восстановите дерево.
- 8.6. Дано  $n$  элементов, для каждого из них известна частота обращений  $f_i$ . Постройте оптимальное дерево поиска, то есть дерево, минимизирующее величину  $\sum f_i \cdot d_i$ , где  $d_i$  — глубина узла  $i$ .
- 8.7. Дано двоичное дерево из  $n$  узлов. Убрать из него минимальное число узлов, чтобы оно стало корректным Scapegoat деревом для данного  $\alpha$ .
- 8.8. Покажите, как на базе дерева поиска сделать мультимножество, то есть хранить множество, в котором может быть несколько объектов с одинаковым ключом.
- 8.9. Есть два двоичных дерева. За одно действие можно поменять местами детей у любого узла. Проверить, что одно дерево можно получить из другого.
- 8.10. Двоичное дерево. Расставить в вершины дерева различные числа длиной  $O(\log n)$  бит так, чтобы по числам в двух данных вершинах можно было (с помощью хеш-таблиц и битовой магии) за  $O(1)$  понять, какое число находится в их ближайшем общем предке.

## 9 LCA, двоичные подъемы

- 9.1. Дано дерево, на каждом ребре написано число. Отвечать на запросы «найти сумму на пути от  $v$  до  $u$ » за  $O(\log n)$  (ну или сразу за  $O(1)$ ).
- 9.2. Дано дерево, на каждом ребре написано число. Отвечать на запросы «найти минимум на пути от  $v$  до  $u$ » (веса не меняются) за  $O(\log n)$ .
- 9.3. Научитесь вычислять любую ассоциативную функцию на пути в дереве за  $O(\log n)$ .
- 9.4. Дано дерево, у каждого ребра есть длина. Научитесь отвечать на запросы: «найти ближайшего предка  $v$ , расстояние до которого не меньше  $l$ » за  $O(\log n)$ .
- 9.5. Дано дерево, у каждого ребра есть длина. Научитесь отвечать на запросы: «найти середину пути от  $v$  до  $u$ » за  $O(\log n)$  (середина пути — это либо вершина, либо точка на ребре, во втором случае можете просто найти нужное ребро).
- 9.6. Запросы: «дано множество из  $k$  вершин, найдите их ближайшего общего предка» за  $O(k \log n)$ .
- 9.7. Запросы: «дано множество из  $k$  вершин, найдите число вершин, которые являются предком хотя бы для одной из них»  $O(k \log n)$ .
- 9.8. Пусть дерево иногда меняется. Добавим две операции: «создать вершину  $v$  и подвесить ее к  $u$ » и «удалить лист  $v$ ». Покажите, что можно пересчитывать двоичные подъемы без потери производительности структуры.
- 9.9. Запросы: «найти расстояние от  $v$  до  $u$ », «поменять вес ребра», оба за  $O(\log n)$ .
- 9.10. Запросы: «найти длину пересечения двух путей» за  $O(\log n)$ .
- 9.11. В дереве покрашены листья, для каждой вершины найти количество различных цветов в поддереве  $O(n \log n)$ .
- 9.12. Запросы: 1) посчитать LCA двух вершин и 2) переподвесить поддерево вершины  $u$  куда-то к другой вершине (все за  $O(\log n)$ ).
- 9.13. Начиная в корне дерева, посетить  $k$  заданных вершин и вернуться в корень. найти минимальную стоимость такого пути за  $O(k \log n)$ .

## 10 Heavy-Light декомпозиция и другие задачи на деревья

- 10.1. Какое максимальное и минимальное число путей может быть в Heavy-light декомпозиции дерева?
- 10.2. Есть другой способ строить Heavy-light декомпозицию: в качестве тяжелых выбирать ребра  $(u, v)$ , для которых  $size(v) \geq size(u)/2$ . Покажите, что такой метод тоже работает и имеет ту же асимптотику. Будет ли он лучше или хуже метода, который выбирает наибольшего ребенка?
- 10.3. Будем выбирать в качестве тяжелого ребра ребро, ведущее в поддерево максимальной высоты (а не максимального веса). Какое максимальное число легких ребер может при этом быть на пути до корня?
- 10.4. Научитесь обрабатывать такие запросы за полилог: 1) изменить вес ребра, 2) найти самый далекий от корня лист.
- 10.5. Есть дерево, каждое ребро может быть либо включено, либо выключено. Научитесь обрабатывать такие запросы за полилог: 1) изменить состояние всех ребер на пути от  $u$  до  $v$ , 2) найти число компонент связности по включенным ребрам.



- 10.6. Есть дерево, каждое ребро может быть либо включено, либо выключено. Научитесь обрабатывать такие запросы за полилог: 1) изменить состояние ребра, 2) найти самый длинный путь от корня по включенным ребрам.
- 10.7. Есть дерево, каждое ребро может покрашено в один из 7 цветов. Научитесь обрабатывать такие запросы за полилог: 1) покрасить все ребра на пути из  $u$  до  $v$  в цвет  $c$ , 2) найти число ребер каждого цвета.
- 10.8. Есть дерево, вершины бывают включены и выключены. Запросы: 1) выключить вершину (обратно не включаются), 2) найти для данной вершины ближайшего включенного предка. Быстрее, чем за  $O(\log n)$ .
- 10.9. Есть дерево дорог, вершина 1 — это столица, дороги бывают хорошие и плохие. Найти минимальное число дорог, которое нужно починить, чтобы на пути от столицы до любой вершины было не более одной плохой дороги  $O(n)$ .
- 10.10. Есть дерево. На каждой вершине было записано число. Для каждой вершины посчитали сумму чисел на соседних вершинах. Восстановите исходные числа за  $O(n)$ .
- 10.11. Было дерево. Для каждой вершины записали два числа: ее степень (число ребер), и  $\widehat{\text{hog}}$  номеров соседних вершин. Восстановите дерево по этим числам за  $O(n)$ .
- 10.12. Дано дерево, в нем выделено  $k$  вершин. Посчитать двоичные подъемы в выделенных вершинах за  $O(k \log n + n)$ .
- 10.13. Множество вершин называется независимым, если никакие две вершины не соединены ребром. Дано дерево, найдите максимальное независимое множество.
- 10.14. Дано дерево. Постройте в нем несколько вершинно непересекающихся путей с максимальной суммарной длиной за  $O(n)$ .
- 10.15. Есть сеть железных дорог в виде дерева из двухколейных путей. Для каждого пути известно время, за которое его проходит поезд. Нужно отвечать на запросы: «Первый поезд отправляется из  $A$  в  $B$  во время  $X$ , второй — из  $C$  в  $D$  во время  $Y$ , правда ли, что есть момент времени, в который поезда едут по одному и тому же ребру?».
- 10.16. Дано неподвешенное дерево. Посчитать для каждой вершины сумму расстояний до всех остальных вершин за  $O(n)$ .
- 10.17. Дерево поджигают в некотором узле, огонь распространяется вдоль ребра за 1 секунду. Посчитайте для каждой вершины, за сколько секунд сгорит дерево, если поджечь в ней.
- 10.18. То же самое, но нужно отвечать на запросы: для заданных двух вершин, если одновременно поджечь в них, за сколько сгорит дерево?

## 11 Link-cut дерево и другие задачи на деревья

- 11.1. Научитесь за  $O(\log n)$  искать LCA в link-cut дереве.
- 11.2. Добавить в link-cut дерево операцию `connected(u, v)`, проверяющую, что вершины  $u$  и  $v$  находятся в одном дереве.
- 11.3. Посмотрите, как меняется потенциал после операций `link` и `cut`, убедитесь, что амортизированное время работы этих операций  $O(\log n)$ .
- 11.4. Link-cut дерево, ребра бывают двух цветов. Проверять, что в пути до корня ребра чередуются.

- 11.5. Link-cut дерево, вершины бывают  $n$  цветов. Проверять, что в пути до корня нет двух соседних вершин одного цвета.
- 11.6. Link-cut дерево, на каждом ребре написан его вес. Находить на пути от  $v$  до корня ближайшее к  $v$  ребро с весом не больше  $d$ .
- 11.7. Попробуйте добавить в link-cut дерево операцию, переподвешивающую дерево за заданную вершину (то есть, все ребра ориентируются так, чтобы заданная вершина стала корнем). (Указание: посмотрите, у каких ребер меняется направление и что происходит с соответствующими выделенными путями, подумайте, как сделать это изменение в splay-дереве).
- 11.8. Есть граф из  $n$  вершин, изначально пустой. Запросы: 1) добавить ребро, 2) проверить, есть ли путь из  $v$  в  $u$ , проходящий по ребрам с весом не больше  $d$ .
- 11.9. Попробуем поддерживать heavy-light декомпозицию дерева после операций link и cut. У скольких ребер может измениться состояние после одной операции?
- 11.10. Можно ли быстро найти все ребра, у которых изменилось состояние? (Чтобы было проще, будем считать, что тяжелое ребро — это то, для которого  $size(u) \geq size(v)/2$ ).
- (11.10.1) Найти ребра, которые были легкими, а стали тяжелыми.
- (11.10.2) Найти ребра, которые были тяжелыми, а стали легкими.
- 11.11. Дано дерево, в каждой вершине записано число. Запросы: 1) дана вершина  $v$  и число  $x$ . Прибавить ко всем потомкам  $u$  вершины  $v$  число  $x - d(u, v)$ , где  $d(u, v)$  — расстояние между вершинами, 2) найти значение числа в вершине. Отвечать на оба запроса за  $O(\log n)$ .
- 11.12. Дана таблица  $d[i, j]$ . Построить такое взвешенное дерево, чтобы расстояние от вершины  $i$  до вершины  $j$  было равно  $d[i, j]$ .
- 11.13. Есть дерево. За одну операцию можно отрезать от него любой лист. Посчитать число различных множеств вершин, которые можно отрезать за  $k$  таких операций на заданном дереве.
- 11.14. Есть дерево. За одну операцию можно отрезать от него любой лист. Посчитать, сколько есть способов совершить  $k$  таких операций на заданном дереве.
- 11.15. Есть дерево из  $n$  вершин. По этому дереву ходят  $n$  фуникулеров, каждый фуникулер движется от вершины  $x_i$  до вершины  $y_i$  и обратно, с остановками в каждой вершине, при этом вершина  $x_i$  — предок вершины  $y_i$ . Вам нужно за  $O(\log n)$  отвечать на запросы: найти минимальное число фуникулеров, чтобы добраться из вершины  $v$  в вершину  $u$ .
- 11.16. Есть дерево из  $n$  вершин. По этому дереву ходят  $n$  автобусов, каждый автобус движется от вершины  $x_i$  до вершины  $y_i$  и обратно, с остановками в каждой вершине. Вам нужно за  $O(\log n)$  отвечать на запросы: есть ли автобус, на котором можно доехать от вершины  $v$  до вершины  $u$ .

## 12 Euler-Tour дерево и другие задачи про деревья

- 12.1. Пусть все вершины имеют веса.
- (12.1.1) узнавать суммарный вес всех вершин в заданной компоненте связности
- (12.1.2) узнавать среднее арифметическое весов всех вершин в компоненте связности
- 12.2. Пусть все вершины покрашены в какие-то цвета.
- (12.2.1) узнавать, правда ли раскраска дерева правильная (все соседние вершины разного цвета)
- (12.2.2) найти любые две соседние вершины одинакового цвета

- (12.2.3) найти число пар соседних вершин одинакового цвета
- 12.3. Есть набор деревьев, нужно делать операции  $\hat{\text{link}}$  и  $\hat{\text{cut}}$ , и узнавать диаметр дерева.
- 12.4. Следующие две задачи относятся к техническим деталям алгоритма проверки динамической связности, который я рассказывал на лекции.
- (12.4.1) В остовном дереве у каждого ребра есть уровень, при этом все уровни не меньше  $L$ , покажите, как перебрать все ребра, уровень которых равен  $L$ , потратив  $O(\log n)$  на каждое ребро.
- (12.4.2) В дереве из каждой вершины есть неостовные ребра (не входящие в дерево), у каждого такого ребра тоже есть уровень, и все уровни не меньше  $L$ . Покажите, как перебрать все ребра, уровень которых равен  $L$ , потратив  $O(\log n)$  на каждое ребро.
- 12.5. У каждой вершины есть вес. Надо уметь увеличить вес всех вершин в компоненте вершины  $v$  на  $d$ , узнать текущий вес вершины  $v$ , узнать суммарный вес компоненты вершины  $v$ .
- 12.6. На каждом ребре написано число. Покажите, как в Euler-Tour дереве узнавать  $\text{xor}$  чисел на пути.
- 12.7. В некоторых задачах возникают графы, в которых в каждой компоненте связности не более одного цикла. Научитесь делать с такими графами все то же самое, что и с Euler-Tour деревьями.
- 12.8. Пусть в каждой компоненте связности выполняется условие:  $E - V \leq C$ , где  $E$  — число ребер,  $V$  — число вершин,  $C$  — небольшая константа. Как работать с такими графами?
- 12.9. В лекциях MIT (<http://courses.csail.mit.edu/6.851/fall17/scribe/L20.pdf>) предлагается хранить Euler-Tour деревья как последовательность вершин (а не ребер), при этом вершина входит несколько раз — каждый раз, когда через нее проходит путь. Для того, чтобы делать операцию  $\text{cut}$  предлагается хранить для каждой вершины самое левое и самое правое вхождение в соответствующем обходе. В чем будет проблема у такого метода?
- 12.10. Дано дерево, на некоторых вершинах написаны числа. Проверить, есть ли обход дерева, при котором первые вхождения помеченных вершины идут в порядке возрастания чисел на них.
- 12.11. Дано дерево, в вершинах записаны числа. Отвечать на запросы: найти число различных чисел на пути от  $u$  до  $v$ , при этом  $u$  — предок  $v$ .

## 13 Центроидная декомпозиция и другие задачи про деревья

- 13.1. Пусть в дереве степени всех вершин не больше 3. Докажите, что есть ребро, при удалении которого дерево распадется на компоненты, размер каждой из которых не больше  $\frac{2}{3}n + 1$ .
- 13.2. Докажите, что в дереве не более двух центроидов.
- 13.3. Задано подвешенное дерево. Найдите для каждого поддерева его центроид. Время  $O(n)$ .
- 13.4. Дано дерево, каждое ребро имеет длину. Научитесь за полилог выполнять операции: 1) поменять длину ребра, 2) найти число вершин на расстоянии не больше  $D$  от заданной.
- 13.5. Дано дерево, каждая вершина имеет положительный вес. Научитесь за полилог выполнять операции: 1) поменять вес вершины, 2) найти путь максимального веса, проходящий через заданную вершину.
- 13.6. Дано дерево, вершины которого можно красить. Отвечать на запросы: по заданным  $(v, d, c)$  покрасить все вершины на расстоянии не больше  $d$  от вершины  $v$  в цвет  $c$ ; узнать, какой сейчас цвет у вершины  $v$ .

- 13.7. 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 больше  $d$  от вершины  $v$ . Отвечать на запросы: провести этап ремонта, узнать, когда последний раз ремонтировалось ребро.
- 13.8. Дано дерево, над ним проводят секретные эксперименты. Каждый эксперимент  $(v, d)$  состоит в том, что в вершине  $v$  распыляют вонючее вещество с вонючестью  $d$ . При этом для всех  $x < d$  уровень вонючести во всех вершинах на расстоянии  $x$  увеличивается на  $d - x$ . Отвечать на запросы: провести эксперимент, найти текущий уровень вонючести в вершине.
- 13.9. Дано дерево, вершины покрашены в черный и белый цвета. Отвечать на запросы: изменить цвет вершины; найти суммарное расстояние от  $v$  до вершин того же цвета.
- 13.10. Дано дерево, вершины покрашены в черный и белый цвета. Отвечать на запросы: изменить цвет вершины; найти ближайшую от  $v$  вершину того же цвета.
- 13.11. Дано дерево путей между городами, для каждого ребра известна стоимость проезда, также известна стоимость колбасы в каждой вершине. Научитесь за полилог искать, как дешевле всего купить колбасу из заданной вершины, с учетом стоимости проезда.
- 13.12. Дано дерево путей между городами, для каждого ребра известно время проезда. Происходят  $n$  событий, про каждое событие известно, в какой вершине и в какое время оно происходит. Нужно узнать, какое максимальное число событий можно посетить. Время  $O(n \log^2 n)$ . (подсказка: сначала придумайте решение за  $O(n^2)$  с помощью динамического программирования)
- 13.13. Дано дерево, в вершинах написаны числа  $c[v]$ . Обработать запросы: изменить значение  $c[v]$ , посчитать сумму  $\sum_u \sum_v c[lca(u, v)]$ . Время  $O(\log n)$ .
- 13.14. Дан массив из чисел. Отвечать на запросы: дан отрезок  $[l, r]$ , найдите максимальное  $k$  такое, что на данном отрезке можно выделить подпоследовательность  $1, 2, \dots, k$ .
- 13.15. Дано дерево, ребра имеют стоимости. Для каждого  $x$  от 0 до  $n - 1$  посчитать минимальную суммарную стоимость ребер, которые нужно удалить, чтобы степени всех вершин были не больше  $x$ . Время  $O(n \log n)$ . (подсказка: сначала придумайте решение для фиксированного  $x$  с помощью динамического программирования)
- 13.16. Рассмотрим клетчатую плоскость, по ней можно перемещаться в соседнюю клетку по стороне. Дана связная фигура из черных клеток без дыр (то есть, множество черных клеток связно, и множество белых клеток связно). Отвечать на запросы: найти длину кратчайшего черного пути между двумя черными клетками. Время  $O(\log n)$ . (подсказка: посчитать отдельно число ходов по вертикали и по горизонтали, для этого разбить фигуру на части таким образом, чтобы они образовали дерево).

## 14 Алгоритмы во внешней памяти

- 14.1. Сделать связный список во внешней памяти. В списке есть указатель на текущий элемент, над ним производятся операции: **next**, **prev**, **add**, **remove**. Время работы  $O(1/B)$ .
- 14.2. Дана матрица  $n \times n$ . Транспонировать ее за  $O(\text{Sort}(n^2))$ .
- 14.3. Дана матрица  $n \times n$ . Известно, что  $M/B > B$ . Транспонировать ее за  $O(n^2/B)$ .
- 14.4. Даны две матрицы  $n \times n$ . Известно, что  $M/B > B$ . Перемножить их за  $O(n^3/B)$ .
- 14.5. Дан массив, найти  $k$ -ю порядковую статистику за  $O(N/B)$ .

- 14.6. Модифицируйте быструю сортировку, чтобы она работала во внешней памяти за оптимальное время ( $O(n/B \log_{M/B} n/B)$ ).
- 14.7. Вычислите значение арифметического выражения со скобками. Время  $O(N/B)$
- 14.8. Есть матрица  $n \times n$ . Найти путь из верхнего левого угла в правый нижний с максимальной суммой (ходить можно только вправо и вниз). Время  $O(n^2/B)$  (найдите только вес пути, можете подумать как восстановить сам путь)
- 14.9. Дерево задано массивом  $p[i]$ , в котором для каждого узла записан номер его родителя. Постройте для каждого узла список его детей за  $O(n/B \cdot \text{polylog}(n))$ .
- 14.10. Дерево задано массивом  $p[i]$ . Постройте двоичные подъемы за  $O(n/B \cdot \text{polylog}(n))$ .
- 14.11. Дерево задано массивом  $p[i]$ . Найдите для каждого узла число узлов в его поддереве за  $O(n/B \cdot \text{polylog}(n))$ .
- 14.12. Дан массив  $next$ . Проверить, что переходы  $(i, next[i])$  образуют один большой цикл за  $O(n/B \cdot \text{polylog}(n))$ .