

Практика по алгоритмам, ВШЭ

Сергей Копелиович, Владислав Кораблинов

Осень, 2020

Contents

1	Асимптотика и линейные алгоритмы	3
1.1	Практика	3
1.2	Домашнее задание	5
1.3	Дополнительные задачи	5
2	Сортировки и кучи	7
2.1	Практика	7
2.2	Домашнее задание	7
2.3	Дополнительные задачи	8
3	Шустрая сортировка и порядковые статистики	13
3.1	Практика	13
3.2	Домашнее задание	14
3.3	Дополнительные задачи	14
4	Демоническое программирование	15
4.1	Практика	15
4.2	Домашнее задание	16
4.3	Дополнительные задачи	16
5	Жадные алгоритмы и динамика	18
5.1	Практика	18
5.2	Домашнее задание	20
5.3	Дополнительные задачи	21
6	Поиск в глубину	22
6.1	Практика	22
6.2	Домашнее задание	22
7	Кратчайшие пути	24
7.1	Практика	24
7.2	Домашнее задание	26
8	Остовные деревья	28
8.1	Практика	28
8.2	Домашнее задание	30
9	Потоки и разрезы	31
9.1	Практика	31
9.2	Домашнее задание	32

10 Бинарные деревья поиска	34
10.1 Практика	34
10.2 Домашнее задание	34
10.3 Дополнительные задачи	35

1 Асимптотика и линейные алгоритмы

1.1 Практика

Напомним определения:

- $f(n) \in \mathcal{O}(g(n)) \equiv \exists N, C > 0 : \forall n \geq N : f(n) \leq C \cdot g(n)$
- $f(n) \in \Omega(g(n)) \equiv \exists N, C > 0 : \forall n \geq N : C \cdot g(n) \leq f(n)$
- $f(n) \in \Theta(g(n)) \equiv \exists N, C_1 > 0, C_2 > 0 : \forall n \geq N : C_1 \cdot g(n) \leq f(n) \leq C_2 \cdot g(n)$
- $f(n) \in o(g(n)) \equiv \forall C > 0 : \exists N : \forall n \geq N : f(n) < C \cdot g(n)$
- $f(n) \in \omega(g(n)) \equiv \forall C > 0 : \exists N : \forall n \geq N : C \cdot g(n) < f(n)$

Все функции здесь $\mathbb{N} \rightarrow \mathbb{N}$ или $\mathbb{N} \rightarrow \mathbb{R}_{>0}$ (далее будет ясно из контекста, какой класс функций используется). В дальнейшем, когда речь идет о принадлежности функций вышеопределенным множествам, мы будем использовать знак “ $=$ ” вместо “ \in ”, т.к. в литературе обычно используются именно такие обозначения.

Асимптотики

1. Докажите, что:

- (a) $f(n) = \Omega(g(n)) \Leftrightarrow g(n) = \mathcal{O}(f(n))$
- (b) $f(n) = \omega(g(n)) \Leftrightarrow g(n) = o(f(n))$
- (c) $f(n) = \Theta(g(n)) \Leftrightarrow f(n) = \mathcal{O}(g(n)) \wedge f(n) = \Omega(g(n))$

2. Контекст имеет значение

Правда ли, что $f(n) = \mathcal{O}(f(n)^2)$?

3. Классы

Определим отношение “ \sim ”. Будем говорить, что $f \sim g$, если $f = \Theta(g)$. Покажите, что \sim — отношение эквивалентности, т.е. оно

- Рефлексивное: $\forall f : f \sim f$,
- Симметричное: $\forall f, g : f \sim g \Leftrightarrow g \sim f$,
- Транзитивное: $\forall f, g, h : (f \sim g) \wedge (g \sim h) \Rightarrow f \sim h$.

4. Порядки

Определим отношение “ \preceq ”. Будем говорить, что $f \preceq g$, если $f = \mathcal{O}(g)$.

Определим отношение $f \preceq g \equiv f = \mathcal{O}(g)$.

- (a) Докажите, что \preceq — отношение предпорядка (рефлексивное и транзитивное)
- (b) Докажите, что \preceq — не отношение частичного порядка, так как не удовлетворяет антисимметричности
- (c) Докажите, что \preceq — отношение частичного порядка на классах эквивалентности по \sim ?

5. Считайте, что функции здесь $\mathbb{N} \rightarrow \mathbb{N}$ и $\forall n : f(n) > 1 \wedge g(n) > 1$.

- (a) $f(n) = \Omega(f(n/2))$?
- (b) $f(n) = \mathcal{O}(g(n)) \Rightarrow \log f(n) = \mathcal{O}(\log g(n))$?
- (c) $f(n) = \mathcal{O}(g(n)) \Rightarrow 2^{f(n)} = \mathcal{O}(2^{g(n)})$?
- (d) $f(n) = o(g(n)) \Rightarrow \log f(n) = o(\log g(n))$?
- (e) $f(n) = o(g(n)) \Rightarrow 2^{f(n)} = o(2^{g(n)})$?
- (f) $\sum_{k=1}^n \frac{1}{k} = \Omega(\log n)$?

6. Определить асимптотику (считайте, что при $x \leq 100$ будет выполняться $T(x) = 100$).

- (a) $T(x) = T(a) + T(x - a) + n$ для натурального числа a .
- (b) $T(x) = T(\frac{x}{2}) + 1$.
- (c) $T(x) = 2 \cdot T(\sqrt{x}) + \log x$

Линейные алгоритмы

7. Дана скобочная последовательность, составленная из скобок '(', ')', '[', ']', '{', '}'. Последовательность называется корректной, если каждой открывающей скобке соответствует закрывающая скобка того же типа, и соблюдается вложенность. Примеры: $(\{\})$ и $()()$ – корректные, а $()$ и $[(])$ – нет.

Придумайте алгоритм, который проверяет корректность последовательности за линейное время.

8. Пусть элементы здесь линейно упорядочены и мы умеем сравнивать их за $\mathcal{O}(1)$.
- (a) Придумайте стек, в котором можно узнавать минимум за $\mathcal{O}(1)$. Все остальные операции стека также должны работать за $\mathcal{O}(1)$.
 - (b) Придумайте очередь, в которой можно узнавать минимум за $\mathcal{O}(1)$. Все остальные операции очереди должны работать за амортизированное $\mathcal{O}(1)$.
 - (c) Придумайте более эффективный по памяти вариант очереди с минимумом на основе пары из обычной очереди и дека.

9. Дан массив целых чисел a_i . Придумайте структуру данных, которая бы умела отвечать на запросы вида “По данным l и r вернуть $\sum_{i=l}^r a_i$ ” за $\mathcal{O}(1)$.

Разрешается сделать предподсчёт за $\mathcal{O}(n)$. Значения в массиве не меняются.

10. Дана последовательность $a_1, a_2, \dots, a_n \in \mathbb{N}$ и $S \in \mathbb{N}$. Найти l, r ($1 \leq l \leq r \leq n$) такие, что сумма $\sum_{i=l}^r a_i = S$. Задачу требуется решить за линейное от n время.
11. Дана последовательность $a_1, a_2, \dots, a_n \in \mathbb{N}$. Для каждого a_i найти самый правый из элементов, которые левее и не больше его. Задачу требуется решить за линейное от n время.
12. Дана последовательность $a_1, a_2, \dots, a_n \in \mathbb{N}$. Найти l, r ($1 \leq l \leq r \leq n$) такие, что
- (a) значение $(r - l + 1) \min_{i \in [l, r]} a_i$ было бы максимально.
 - (b) значение $\left(\sum_{i \in [l, r]} a_i \right) \min_{i \in [l, r]} a_i$ было бы максимально.

Задачу требуется решить за линейное от n время.

13. Вам дан массив натуральных чисел и число k . Требуется найти подотрезок массива такой, что НОК чисел на нем равен k или заявить, что такого нет. Время работы: $\mathcal{O}(nT_{LCM}(k))$, где $T_{LCM}(k)$ — время подсчета НОК для чисел размера k .

1.2 Домашнее задание

- Дайте ответ для двух случаев $\mathbb{N} \rightarrow \mathbb{N}$ и $\mathbb{N} \rightarrow \mathbb{R}_{>0}$:
 - Если в определении \mathcal{O} опустить условие про N (т.е. оставить просто $\forall n$), будет ли полученное определение эквивалентно исходному?
 - Тот же вопрос про o .
- Считайте здесь, что $\forall n : f(n) > 1 \wedge g(n) > 1$. Правда ли, что $f(n) = o(g(n)) \Rightarrow 2^{f(n)} = o(2^{g(n)})$?
- Заполните табличку и поясните (особенно строчки 4 и 7):

A	B	\mathcal{O}	o	Θ	ω	Ω
n	n^2	+	+	—	—	—
$\log^k n$	n^ϵ					
n^k	c^n					
\sqrt{n}	$n^{\sin n}$					
2^n	$2^{n/2}$					
$n^{\log m}$	$m^{\log n}$					
$\log(n!)$	$\log(n^n)$					

Здесь все буквы, кроме n , — константы.

- Дана последовательность $a_1, a_2, \dots, a_n \in \mathbb{N}$ и $S \in \mathbb{N}$. Найти l, r ($1 \leq l \leq r \leq n$) такие, что сумма $\sum_{i=l}^r a_i = S$. Задачу требуется решить за линейное от n время. **Подсказки:**
 - Для каждого i найдите максимальное такое r_i , что $\sum_{j=i}^{r_i} a_j \leq S$ за $\mathcal{O}(n)$ для каждого i .
 - Найдите за $\mathcal{O}(n)$ ответ задачи, если известны r_1, \dots, r_n .
 - Докажите, что $r_i \leq r_{i+1}$.
 - Пользуясь предыдущим пунктом найдите все r_i за $\mathcal{O}(n)$.
- Дана последовательность $a_1, a_2, \dots, a_n \in \mathbb{N}$.
 - За $\mathcal{O}(n)$ для каждого a_i найти самый правый из элементов, которые левее и меньше его.
 - За $\mathcal{O}(n)$ для каждого a_i найти самый левый из элементов, которые правее и меньше его.
 - За $\mathcal{O}(n)$ найти l, r ($1 \leq l \leq r \leq n$) такие, что значение $(r - l + 1) \min_{i \in [l, r]} a_i$ было бы максимально.
 - За $\mathcal{O}(n)$ найти l, r ($1 \leq l \leq r \leq n$) такие, что значение $\left(\sum_{i \in [l, r]} a_i\right) \min_{i \in [l, r]} a_i$ было бы максимально.
- Вам дан массив из n элементов и список из m запросов $add(x, l, r)$: прибавить x к каждому элементу на отрезке $[l, r]$. За $\mathcal{O}(n + m)$ выведите массив, получающийся из исходного после выполнения заданных запросов.
- (только группа Антона) Определить асимптотику $T(n) = 2 \cdot T(\lfloor \log n \rfloor) + 2^{\log^* n}$, где $\log^* n$ — **итерированный логарифм**.

1.3 Дополнительные задачи

- Упорядочите функции по скорости роста и обозначьте неравенства между соседями. Укажите, в каких неравенствах $f = o(g)$, а в каких $f = \Theta(g)$

$\log(\log^* n)$	$2^{\log^* n}$	$(\sqrt{n})^{\log n}$	n^2	$n!$	$(\log n)!$
$(3/2)^n$	n^3	$\log^2 n$	$\log n!$	2^{2^n}	$n^{1/\log n}$
$\ln \ln n$	$\log^* n$	$n \cdot 2^n$	$n^{\log \log n}$	$\ln n$	1
$2^{\ln n}$	$(\log n)^{\log n}$	e^n	$4^{\log n}$	$(n+1)!$	$\sqrt{\log n}$
$\log^* \log n$	$2^{\sqrt{2 \log n}}$	n	2^n	$n \log n$	$2^{2^{n+1}}$

Примечание: $\log^*(n) = \begin{cases} 0 & \text{если } n \leq 1; \\ 1 + \log^*(\log n) & \text{иначе.} \end{cases}$

2. Определить асимптотику (считайте, что при $n \leq 100$ будет выполняться $T(n) = 100$).
 - (a) $T(n) = 2 \cdot T(\lfloor \frac{n}{2} \rfloor) + 17 + n$.
 - (b) $T(n) = T(\alpha \cdot n) + T((1 - \alpha) \cdot n) + n$ для произвольной константы $\alpha \in (0, 1)$.
 - (c) $T(n) = 4 \cdot T(\lfloor \frac{n}{2} \rfloor) + n^k$ для $k \in \{1, 2, 3\}$.
3. Дана последовательность $a_1, a_2, \dots, a_n \in \mathbb{Z}$. Найти l, r ($1 \leq l \leq r \leq n$) такие, что сумма $\sum_{i=l}^r a_i$ была бы максимальной. Задачу требуется решить за линейное от n время.
4. Дано число, представленное n цифрами в d -ичной записи без ведущих нулей. Из числа требуется вычеркнуть ровно k цифр так, чтобы результат был максимальным. Задачу требуется решить за линейное от n время.
5. Вам дан массив из n элементов и число k . Все числа лежат в отрезке $[1..n]$. Найдите такие l и r , что на отрезке $[l, r]$ встречается хотя бы k различных элементов, или сообщите, что такого отрезка нет. Если таких отрезков несколько, выберите тот из них, длина которого минимальна. Время работы $\mathcal{O}(n)$.
6. Вам дан массив натуральных чисел и число k . Требуется найти подотрезок массива такой, что НОК чисел на нем равен k или заявить, что такого нет. Время работы: $\mathcal{O}(nT_{LCM}(k))$, где $T_{LCM}(k)$ — время подсчета НОК для чисел размера k .
7. Дана квадратная матрица из нулей и единиц. Найти наибольший по площади подпрямоугольник, состоящий только из нулей за $\mathcal{O}(n^2)$.
8. Вам каждый день на протяжении некоторого времени поступает запрос «вырастет ли курс Apple на бирже», и у вас есть n советников, с которыми вы можете консультироваться. Вы отвечаете да или нет, и в конце каждого дня вам говорят, правильно ли вы ответили. Придумайте алгоритм, который сделает не более $10(\log n + m)$ ошибок, где m — число ошибок, которое сделает лучший советник (подсказка: назначьте советникам веса и изменяйте их в зависимости от правильности их ответов).
9. Придумайте расширяющийся массив с реальным (не амортизированным) временем добавления $\mathcal{O}(1)$.
10. Дан массив целых чисел от 1 до n длины $n + 1$, который нельзя модифицировать. Используя $\mathcal{O}(\log n)$ битов дополнительной памяти, найдите в массиве пару одинаковых чисел за $\mathcal{O}(n)$.
11. Дана последовательность $\sigma = \langle a_1, a_2, \dots, a_m \rangle$, где каждый $a_i \in [n] = \{1, 2, \dots, n\}$. Обозначим частоту появления элемента x через $f_\sigma[x] = |\{i | a_i = x\}|$. Известно, что $\exists_x f_\sigma[x] = 1$ и для всех остальных значений $y \neq x, f_\sigma[y] \equiv 0 \pmod 2$. Требуется найти x за один проход по последовательности, используя $\mathcal{O}(\log n + \log m)$ бит памяти.
12. Дана последовательность $\sigma = \langle a_1, a_2, \dots, a_m \rangle$, где каждый $a_i \in [n]$. Известно, что $\exists_x f_\sigma[x] > \frac{m}{2}$. Требуется найти x за один проход по последовательности, используя $\mathcal{O}(\log n + \log m)$ бит памяти.

2 Сортировки и кучи

2.1 Практика

1. Есть n веревок, каждая имеет целую длину l_i , которые можно резать. Нужно получить k одинаковых кусков максимальной целочисленной длины (также могут остаться неиспользованные обрезки). $\mathcal{O}(n \log l_{\max})$.
2. Есть k отсортированных массивов. В сумме массивы содержат n элементов. Слить массивы за $\mathcal{O}(n \log k)$.
3. Придумайте детерминированную структуру данных на основе бинарной кучи, которая умеет делать `Insert(x)`, `DeleteMedian()`, все операции за $\mathcal{O}(\log n)$.
4. Модифицируйте операцию `SiftUp` для бинарной кучи так, чтобы она по-прежнему работала за $\mathcal{O}(\log n)$, но при этом делала лишь $\mathcal{O}(\log \log n)$ сравнений.
5. Модифицируйте операцию `SiftDown` для бинарной кучи так, чтобы она по-прежнему работала за $\mathcal{O}(\log n)$, но при этом делала лишь $\log_2 n + \mathcal{O}(\log \log n)$ сравнений.
6. Даны два массива a и b длины n , сгенерировать все попарные суммы $a_i + b_j$ в отсортированном порядке.

(a) За $\mathcal{O}(n^2 \log n)$.

(b) За $\mathcal{O}(n^3)$ с использованием $\mathcal{O}(n)$ дополнительной памяти.

(c) За $\mathcal{O}(n^2 \log n)$ с использованием $\mathcal{O}(n)$ дополнительной памяти.

(d) За $\mathcal{O}(n^3)$ с использованием $\mathcal{O}(1)$ дополнительной памяти.

Здесь считайте, что дополнительная память — количество чисел длины $\mathcal{O}(\log n)$, которые вы можете сохранить.

7. Покажите, что любая сортировка сравнениями, которая верно работает хотя бы на доле $\frac{1}{100^n}$ от всех перестановок, не может работать за $\mathcal{O}(n \log n)$ на всех тестах.
8. Дана обычная бинарная куча (с минимумом в голове), требуется узнать k -й минимум.
 - (a) $\mathcal{O}(k \log n)$
 - (b) $\mathcal{O}(k^2)$
 - (c) $\mathcal{O}(k \log k)$
9. Пусть дан массив размера n из целых чисел. Требуется сделать предварительные вычисления (в будущем мы будем кратко называть их *предподсчёт*) за $\mathcal{O}(n \log n)$, чтобы затем ответить на некоторое неизвестное число запросов про массив вида “сколько раз число x встречается на отрезке $[l..r]$ ”, причём на каждый запрос можно потратить $\mathcal{O}(\log n)$ времени.
10. Инверсией в массиве чисел $a[\dots]$ называется такая пара индексов i, j , что $i < j$, но $a_i > a_j$.

Дан массив из n различных элементов. Требуется найти число инверсий за $\mathcal{O}(n \log n)$.

2.2 Домашнее задание

1. Есть m стойл с координатами x_1, \dots, x_m и n коров. Расставить коров по стойлам (не более одной в стойло) так, чтобы минимальное расстояние между коровами было максимально. $\mathcal{O}(m(\log m + \log x_{\max}))$.
2. (только группа Антона) Даны два отсортированных массива длины n , которые нельзя модифицировать. Найдите k -ю порядковую статистику в объединении массивов (то есть элемент, находившийся бы на k -ой позиции если бы массивы слили), используя $\mathcal{O}(1)$ дополнительной памяти.
 - (a) За $\mathcal{O}(\log^2 n)$.

- (b) За $\mathcal{O}(\log n)$.
3. (только группа Влада) Даны два массива a и b длины n , сгенерировать все попарные суммы $a_i + b_j$ в отсортированном порядке.
- (a) За $\mathcal{O}(n^2 \log n)$.
 (b) За $\mathcal{O}(n^3)$ с использованием $\mathcal{O}(n)$ дополнительной памяти.
 (c) За $\mathcal{O}(n^2 \log n)$ с использованием $\mathcal{O}(n)$ дополнительной памяти.
 (d) За $\mathcal{O}(n^3)$ с использованием $\mathcal{O}(1)$ дополнительной памяти.
- Здесь считайте, что дополнительная память — количество чисел длины $\mathcal{O}(\log n)$, которые вы можете сохранить.
4. (только группа Влада) Дана обычная бинарная куча (с минимумом в голове), требуется узнать k -й минимум.
- (a) $\mathcal{O}(k \log n)$
 (b) $\mathcal{O}(k \log k)$
5. (только группа Влада) Пусть дан массив размера n из целых чисел. Требуется сделать предварительные вычисления (в будущем мы будем кратко называть их *предподсчёт*) за $\mathcal{O}(n \log n)$, чтобы затем ответить на некоторое неизвестное число запросов про массив вида “сколько раз число x встречается на отрезке $[l..r]$ ”, причём на каждый запрос можно потратить $\mathcal{O}(\log n)$ времени.
6. В свободное время Анка-пулемётчица любит сортировать патроны по серийным номерам. Вот и сейчас она только разложила патроны на столе в строго отсортированном порядке, как Иван Васильевич распахнул дверь с такой силой, что все патроны на столе подпрыгнули и немного перемешались. Оставив ценные указания, Иван Васильевич отправился восвояси. Как оказалось, патроны перемешались не сильно. Каждый патрон отклонился от своей позиции не более чем на k . Всего патронов n . Помогите Анке отсортировать патроны.
- (a) Отсортируйте патроны за $\mathcal{O}(nk)$.
 (b) Отсортируйте патроны за $\mathcal{O}(n + I)$, где I — число инверсий.
 (c) Докажите нижнюю оценку на время сортировки $\Omega(n \log k)$.
 (d) Отсортируйте патроны за $\mathcal{O}(n \log k)$ (если вы решили этот пункт, автоматически засчитается и пункт а).
7. (только группа Антона) Дано $2n - 1$ коробок с чёрными и белыми шарами. В i -ой коробке находится w_i белых и b_i чёрных шаров. Всего в коробках находится W белых и B чёрных шаров. Требуется выбрать n коробок таким образом, чтобы суммарное число белых шаров в них было не менее $\frac{W}{2}$, а чёрных не менее $\frac{B}{2}$. Решить за $\mathcal{O}(n \log n)$.

2.3 Дополнительные задачи

1. Дан массив длины n , в котором встречаются $m \leq n$ различных элементов.
- (a) Пусть зафиксирован набор частот элементов $p_i > 0, i = 1 \dots m$. Докажите нижнюю оценку $n \left(\sum_{i=1}^m p_i \log \frac{1}{p_i} \right) - n \log e$ на число сравнений в худшем случае при сортировке сравнениями. Полезный факт: $n \ln n \geq \ln n! = n \ln n - n + \mathcal{O}(\ln n)$.
 (b) Докажите нижнюю оценку $n \log \frac{m}{e}$ на число сравнений в худшем случае при сортировке сравнениями в пределе, если $m = o(n)$.
2. Куча хранится в массиве длины n . Родитель p хранит детей в ячейках $2 \cdot p + 1$ и $2 \cdot p + 2$. Алгоритм приступает к сортировке. Сортировка устроена следующим образом.
- Поменять первый и последний элемент кучи местами.

- Уменьшить размер кучи на единицу.
- Запустить **SiftDown** на первом элементе.

SiftDown меняет родителя с наибольшим ребенком (при условии, что ребенок больше родителя) и запускается рекурсивно. Требуется придумать алгоритм, который по n выдаёт перестановку чисел от 1 до n , которая является корректной кучей и приводит к максимальному количеству вызовов **SiftDown** при сортировке. Время работы — $\mathcal{O}(n \log n)$.

- Дано множество из n точек на плоскости. Найти пару ближайших точек за $\mathcal{O}(n \log n)$.
 - Дано множество из n векторов на плоскости. Разрешается координату любого вектора умножить на -1. Найти пару векторов, чья сумма минимальна, за $\mathcal{O}(n \log n)$.
- Дано бинарное дерево: $Tree ::= Node(Tree, Tree) | Empty$ (эта запись означает, что дерево — это либо вершина с парой потомков-деревьев, либо особое значение *Empty*). Определим функцию **rank**(x) следующим образом:

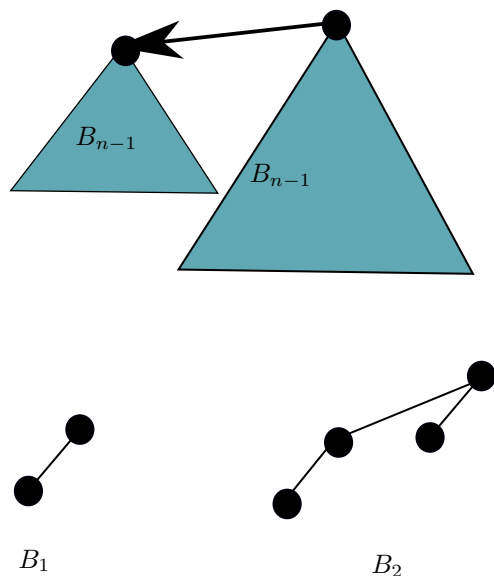
- **rank**(*Empty*) = 0
- **rank**($Node(left, right)$) = $\min(\mathbf{rank}(left), \mathbf{rank}(right)) + 1$.

Назовём бинарное дерево *скошенным влево (левацким)*, если для его вершин выполнено следующее свойство:

$$\forall x = Node(left, right) \mathbf{rank}(left) \geq \mathbf{rank}(right).$$

Скошенная влево (левацкая) куча — это скошенное влево дерево, в вершинах которого хранятся данные, для которых выполнено свойство кучи.

- Докажите, что для любого скошенного влево дерева $|T| \geq 2^{\mathbf{rank}(T)} - 1$ ($|T|$ обозначает количество вершин в дереве T).
 - Придумайте, как слить две скошенные влево кучи H_1 и H_2 за время $\mathcal{O}(\log |H_1| + \log |H_2|)$.
 - Придумайте, как используя операцию слияния, построенную на предыдущем шаге, реализовать операции:
 - **Insert**(x) — добавление элемента x в кучу,
 - **ExtractMin**() — удаление минимального элемента из кучи.
- Пусть B_n (биномиальное дерево порядка n) определено следующим образом:
 - при $n = 0$ это дерево из одной вершины.
 - при $n > 0$ это B_{n-1} , корню которого первым ребенком подвешено еще одно B_{n-1} .



- (a) Докажите, что B_n имеет высоту n .
 - (b) Докажите, что в B_n содержится ровно 2^n вершин.
 - (c) Определим биномиальную кучу как набор биномиальных деревьев, в котором нет двух деревьев одного порядка. Покажите, что для любого n существует биномиальная куча с n вершинами.
 - (d) Пусть на всех деревьях биномиальной кучи выполняется свойство кучи (min в голове). Придумайте **GetMin** за $\mathcal{O}(\log n)$.
 - (e) Придумайте **Merge** за $\mathcal{O}(\log n)$.
 - (f) Придумайте **Add** за $\mathcal{O}(\log n)$.
 - (g) Придумайте **ExtractMin** за $\mathcal{O}(\log n)$.
 - (h) Придумайте **DecreaseKey** по ссылке на узел. $\mathcal{O}(\log n)$.
 - (i) Придумайте **Delete** по ссылке на узел. $\mathcal{O}(\log n)$.
6. Покажите, что n операций **Add** подряд в биномиальную кучу работают за $\mathcal{O}(n)$.
7. Рассмотрим бинарную скошенную систему исчисления. На каждой позиции в скошенной записи числа может стоять цифра 0, 1 или 2. Число $\overline{a_k a_{k-1} \dots a_2 a_1}$ в скошенной системе переводится в десятичную по формуле $\sum_{i=1}^k a_i \cdot (2^i - 1)$.
 В скошенной системе счисления есть два ограничения: цифра 2 может встречаться в записи не более одного раза; все цифры следующих меньших разрядов равны нулю. Пример первых чисел: 1, 2, 10, 11, 12, 20, 100, 101...
- (a) Докажите, что каждое неотрицательное целое число имеет единственное возможное представление в скошенной системе счисления.
 - (b) Придумайте, как увеличить число в скошенной системе на единицу за $\mathcal{O}(1)$.
8. Определим структуру данных “скошенный список”. Список длины n строится так:
- запишем число n в скошенной системе счисления: $\overline{a_k a_{k-1} \dots a_2 a_1}$

- для каждого i смотрим в соответствующую позицию скошенной записи числа n и создаём a_i полных двоичных деревьев высоты i
- размещаем n элементов списка: сперва выбираем дерево в порядке возрастания высоты, а внутри конкретного дерева размещаем в порядке обхода в глубину: «корень, левый ребёнок, правый ребёнок»

Примеры скошенных списков длин 1, 2, 3, 4, 5:

Figure 1: Лист [a] (число: 1)



Figure 2: Лист [a b] (число: 2)



Figure 3: Лист [a b c] (число: 10)

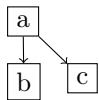


Figure 4: Лист [a b c d] (число: 11)

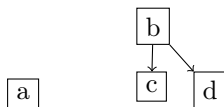
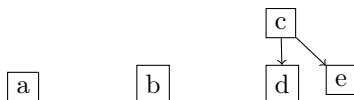


Figure 5: Лист [a b c d e] (число: 12)



Придумайте, как реализовать следующие операции со списком длины n :

- Добавление элемента в начало списка за $\mathcal{O}(1)$.
- Доступ к i -му элементу за $\mathcal{O}(\log n)$.
- Получить скошенный список из k последних элементов данного скошенного списка за $\mathcal{O}(\log n)$.

9. Придумайте структуру данных, которая поддерживает следующие операции (в оценках времени работы n — текущее количество элементов):

- **Insert(x)** — добавление элемента x за $\mathcal{O}(\log n)$,
- **ExtractMin()** — удаление минимального элемента за $\mathcal{O}(\log n)$,
- **Clone()** — копирование структуры за $\mathcal{O}(1)$ (после копирования с каждой из копий можно независимо проделывать любую из данных трех операций).

Подсказка: за основу можно взять левацкую кучу.

10. Структура данных «файл последовательного доступа» поддерживает следующие операции за $\mathcal{O}(1)$:

- *Read()*: чтение числа из файла на текущей позиции и перевод позиции вперёд на 1 элемент.
- *Write(x)*: запись числа в файл в текущую позицию и перевод позиции вперёд на 1 элемент.
- *Rewind()*: перевод позиции на начало файла.

Требуется отсортировать файл за $\mathcal{O}(n \log n)$ используя $\mathcal{O}(1)$ дополнительной памяти и $\mathcal{O}(1)$ дополнительных файлов.

3 Шустрая сортировка и порядковые статистики

3.1 Практика

1. (а) Приведите вероятностный алгоритм поиска медианы в массиве различных чисел со средним временем работы $\mathcal{O}(n)$
(б) Приведите детерминированный алгоритм поиска медианы в массиве различных чисел с гарантированным временем работы $\mathcal{O}(n)$
2. Придумайте, как добиться от QuickSort времени $\mathcal{O}(n \log n)$ в худшем случае.
3. Робот Иван Семеныч пробует пирожки. Содержимое пирожков делится на три типа. Всего пирожков n . Каждый пирожок можно попробовать не более одного раза. Любые два пирожка можно поменять местами. Память у робота маленькая, $\mathcal{O}(\log n)$ бит. Помогите Ивану Семенычу отсортировать пирожки по типу: сначала первый, потом второй, потом третий. Сортировка должна работать за линейное время.
4. Дан набор из n пар гаек и болтов, в разных парах размеры гаек и болтов различны. Гайки и болты перемешаны. Требуется для каждой гайки найти соответствующий болт. Сравнивать можно только болты с гайками (сравнить две гайки между собой, или два болта между собой — невозможно). $\mathcal{O}(n \log n)$ в среднем.
5. Пусть задан массив A из $n = a \cdot k$ различных чисел. Требуется разбить массив на k частей по a элементов в каждой так, чтобы любой элемент части i был бы меньше любого элемента части $i + 1$ ($\forall i \in [1, k - 1]$). $\mathcal{O}(n \log k)$ в среднем.
6. Дан массив из $2 \cdot n - 1$ числа, который нельзя модифицировать. Есть дополнительная память на $n + 1$ элемент массива и ещё $\mathcal{O}(1)$ сверху. Требуется найти медиану за $\mathcal{O}(n \log n)$.
7. Дана последовательность из n чисел, нужно за один проход и $\mathcal{O}(n)$ времени в среднем найти в ней k минимумов, используя $\mathcal{O}(k)$ дополнительной памяти.
8. Даны массив из n чисел и m чисел p_1, p_2, \dots, p_m , нужно за $\mathcal{O}(n \log m + m)$ для каждого i найти p_i -ую порядковую статистику.
9. Дан массив из $2n$ чисел. Найти минимальное и максимальное за $3n - 2$ сравнения.
10. Найти второй максимум в массиве за $n + \mathcal{O}(\log n)$ сравнений.

3.2 Домашнее задание

1. Оцените время работы детерминированного алгоритма поиска порядковой статистики, если вместо пятерок разбивать элементы на
 - (a) семерки.
 - (b) тройки.
2. (*только группа Влада*) Даны массив из n чисел и m чисел p_1, p_2, \dots, p_m , нужно за $\mathcal{O}(n \log m + m)$ для каждого i найти p_i -ую порядковую статистику.
3. Дан массив $A[1..n]$ из n различных чисел. Массив не обязательно отсортирован. Требуется найти k ближайших к медиане элементов за линейное время. Решить для двух метрик.
 - (a) По позиции в отсортированном массиве.

$$d(x, \text{median}) = |\text{pos}(x) - \text{pos}(\text{median})|,$$

где $\text{pos}(x)$ — позиция элемента x в отсортированном массиве.

- (b) По значению.

$$d(x, \text{median}) = |x - \text{median}|.$$

4. Даны два массива из положительных целых чисел a и b , размер обоих равен n . Выбрать массив p из k различных чисел от 1 до n так, чтобы $\frac{\sum_{i=1}^k a_{p_i}}{\sum_{i=1}^k b_{p_i}} \rightarrow \max$. Время $\mathcal{O}(n \log M)$, где $M = \max(\max(a_i), \max(b_i), n)$.
5. Докажите, что для поиска максимума в массиве различных чисел потребуется как минимум $n - 1$ сравнение.
6. Дана последовательность из n чисел, нужно за один проход и $\mathcal{O}(n)$ времени в среднем найти в ней k минимумов, используя $\mathcal{O}(k)$ дополнительной памяти.
7. (*только группа Антона*) Дан массив из $2n$ различных чисел. Найдите минимальное и максимальное за $3n - 2$ сравнения и докажите, что это точная нижняя оценка, то есть меньшего количества сравнений может не хватить.

3.3 Дополнительные задачи

1. В матрице Q из натуральных чисел размера $N \times N$ найти подматрицу размера $H \times W$ с максимальной медианой. H, W — нечётные.
 - (a) $\mathcal{O}(N^2 \log Q_{\max})$. Здесь Q_{\max} — максимальный элемент матрицы.
 - (b) $\mathcal{O}(N^2 \log N)$.
2. Пусть алгоритм A находит i -ый по порядку элемент, используя только попарные сравнения элементов. Покажите, что, используя результаты только этих сравнений, можно найти все элементы, меньшие i -ого, и все элементы, большие i -ого.
3. Дан массив из $2n$ различных чисел. Найдите минимальное и максимальное за $3n - 2$ сравнения и докажите, что это точная нижняя оценка, то есть меньшего количества сравнений может не хватить.
4. Найдите второй максимум в массиве за $n + \lceil \log_2 n \rceil - 2$ сравнение и докажите, что это точная нижняя оценка, то есть меньшего количества сравнений может не хватить.
5. Дан массив длины n . Изначально выделен отрезок позиций $1 \dots d$. Далее $n - d$ раз поступает команда «выведите медиану чисел в окне и сдвиньте отрезок на 1 направо».
 - (a) Обработайте каждую команду за $\mathcal{O}(\log n)$.
 - (b) Докажите, что не существует такой функции $f(n) \in o(\log n)$, что каждую команду можно обработать за $f(n)$.

4 Демоническое программирование

4.1 Практика

1. Найдите максимальную возрастающую подпоследовательность за $\mathcal{O}(n \log n)$.

- a) Найти длину
- b) Восстановить ответ

2. Найти максимальное по весу паросочетание за $\mathcal{O}(n)$ на

- (a) дереве из n вершин,
- (b) простом цикле из n вершин,
- (c) связном неориентированном графе из n вершин и n рёбер.

Веса на рёбрах.

3. Есть следующее рекуррентное соотношение:

$$\begin{aligned}a_n &= a_{n-1} + 2c_{n-1} + 1 \\b_n &= 5 - c_{n-1} \\c_n &= c_{n-2} - b_{n-1}\end{aligned}$$

Нам известны $a_0, a_1, b_0, b_1, c_0, c_1$. Найти a_n, b_n, c_n по модулю p за $\mathcal{O}(\log n)$.

4. Даны две последовательности длины n . Придумайте, как найти наидлиннейшую общую подпоследовательность этих последовательностей.

- (a) За $\mathcal{O}(n^2)$.
- (b) За $\mathcal{O}(n \log n)$, в случае, если в одной из последовательностей все элементы различны.

5. Дан массив из n целых чисел и число d . Найти подпоследовательность максимальной длины с условием, что соседние элементы в ней должны отличаться не более чем на d .
6. Дан массив из n целых чисел, число d и число k . Найти подпоследовательность длины k с максимальной суммой элементов при условии, что соседние элементы в ней должны отличаться не более чем на d .
7. Дано натуральное число $s \leq 300$. Найти набор натуральных чисел, сумма которых будет равна s , а их наименьшее общее кратное — максимально.

4.2 Домашнее задание

1. Дана строка s длины n . Для каждой пары (i, j) найти длину максимального общего префикса i -го и j -го суффиксов строки s . $\mathcal{O}(n^2)$.
2. Дан набор нечестных монеток с вероятностью выпадения орла p_1, p_2, \dots, p_n . Требуется посчитать вероятность выпадения ровно k орлов за $\mathcal{O}(n \cdot k)$. Операции над числами считать выполнимыми за $\mathcal{O}(1)$.
3. (только группа Влада)
Дан массив из n целых чисел и число d . Найти подпоследовательность максимальной длины с условием, что соседние элементы в ней должны отличаться не более чем на d за $\mathcal{O}(n^2)$.
4. (только группа Влада)
Дан массив из n целых чисел, число d и число k . Найти подпоследовательность длины k с максимальной суммой элементов при условии, что соседние элементы в ней должны отличаться не более чем на d . $\mathcal{O}(n^2 k)$.
5. (только группа Серёжи)
Пусть есть n подарков разной натуральной стоимости и три поросёнка. Нужно раздать подарки как можно честнее (так, чтобы минимизировать разность суммарной стоимости подарков самого везучего поросёнка и самого невезучего). Придумайте алгоритм решения данной задачи за $\mathcal{O}(nW^2)$, где W — суммарная стоимость подарков.
6. Клетки поля $n \times 5$ покрашены в чёрный и белый цвета. Будем называть получившийся узор красивым, если он не содержит одноцветного квадрата 2×2 . Вычислите число красивых узоров по модулю небольшого простого числа за время $\mathcal{O}(\log n)$.

4.3 Дополнительные задачи

1. (1 балл за пункты $a + b$ вместе и 1 балл за пункт c)
Дана строка из латинских букв длины n , нужно ее запаковать в максимально короткую, используя правило (k, i) — повторить k символов начиная с i -й позиции. Заметим, что длина (k, i) — не константа. Например, $xyababababz \rightarrow xyab(8, 2)z$, $xyaaaaabaaaaab \rightarrow xyab(3, 2)b(10, 2)$ (но это не оптимально, оптимально $xyaaaaab(10, 2)$).
 - (a) $\mathcal{O}(n^3)$.
 - (b) $\mathcal{O}(n^2)$, считая, что длина строки (k, i) — константа.
 - (c) $\mathcal{O}(n^2)$.
2. (1 балл за пункты $a + b$ вместе и 1 балл за пункт c)
Есть k грузовиков с заданной вместимостью, задача — перевезти n вещей с заданными весами минимальным числом заездов. Один заезд — погрузить и отправить все грузовики.
 - (a) $k = 1$, $\mathcal{O}(3^n)$.
 - (b) $k = 2$, $\mathcal{O}(4^n)$.
 - (c) $\mathcal{O}(3^k n)$.
3. (1 балл за пункты $a + b$ вместе и 1 балл за пункт c)
Вычислите, сколькими способами можно замостить доминошками клетчатое поле
 - (a) $n \times 3$, за время $\mathcal{O}(n)$.
 - (b) $n \times t$, за время $\mathcal{O}(4^n t)$.
 - (c) $n \times t$, за время $\mathcal{O}(2^n n t)$.

Ответ посчитать по модулю небольшого простого числа.

4. Вам дана доска фанеры размера $n \times t$. В неё было вбито несколько гвоздей с целыми координатами (от них остались некрасивые дырки). Сколькими способами можно разрезать доску на прямоугольники с целыми сторонами так, чтобы ни один из гвоздей не попал внутрь прямоугольника. Время: $\mathcal{O}(n^2 4^m)$.
5. * (*Эту задачу можно сдавать только устно*)
Приведите полиномиальный алгоритм, вычисляющий количество разбиений клетчатой доски $n \times t$ на доминошки (другими словами, на прямоугольники 1×2 и 2×1).
6. Посчитать по модулю небольшого простого числа количество способов, которыми можно расставить на доске $n \times n$ сколько-либо небьющих друг друга коней, за $\mathcal{O}(n 8^n)$

5 Жадные алгоритмы и динамика

5.1 Практика

1. Подпоследовательность-палиндром

Дана строка длины $n \leq 100$.

Найти максимальную по длине подпоследовательность, которая является палиндромом.

2. Пираты

Судно атакуют пираты. Для каждого пирата известны его азимут a_i и время t_i , через которое пират приплывёт и совершит непотребство. Однако, у судна есть лазерная пушка, которой оно защищается. У пушки есть начальный азимут a и угловая скорость вращения ω . Пушка уничтожает все объекты, на которые она сейчас направлена. Помогите судну определить порядок уничтожения пиратов за $\mathcal{O}(n^2)$, чтобы не допустить непотребства.

3. И снова подпоследовательности

Дан массив из n натуральных чисел: a_1, \dots, a_n . Выберите подпоследовательность $i_1 \leq \dots \leq i_k \in \{1, \dots, n\}$, такую, что $l \leq |i_j - i_{j-1}| \leq r$ и $\sum_{j=1}^k a_{i_j} \rightarrow \max$.

a) За $\mathcal{O}(n^2)$.

b) За $\mathcal{O}(n)$.

4. Subset cover

Дано число n и семейство из m множеств, каждое из которых — подмножество $U = \{1, 2, \dots, n\}$.

Выбрать минимальное количество элементов семейства, чтобы всё U было ими покрыто.

5. Перевозка товаров

Есть грузовик с заданной вместимостью, задача — перевезти n вещей с заданными весами со склада в магазин минимальным числом заездов.

a) $\mathcal{O}(3^n)$. Обязательно.

b) (*) $\mathcal{O}(2^n n)$.

6. Мягкие дедлайны

В фирму поступило n заказов, у каждого есть время исполнения t_i и жёсткий дедлайн d_i . В каком порядке выполнять заказы, чтобы всё успеть?

7. (*) Жёсткие дедлайны

В фирму поступило n заказов, у каждого есть время исполнения t_i и жёсткий дедлайн d_i . В каком порядке и **какие** выполнять заказы, чтобы успеть **побольше**?

8. Заказы и время конца

В фирму поступило n заказов, которые можно выполнять в произвольном порядке. На выполнение заказа i необходимо время t_i . В каждый момент времени можно работать ровно над одним заказом. Пусть e_i — момент окончания выполнения заказа номер i . Распределите работу над заказами так, чтобы минимизировать $\sum_i e_i$. Время $\mathcal{O}(n \log n)$.

9. Атлеты

n атлетов хотят выстроить из своих тел башню максимальной высоты. Башня это цепочка атлетов, первый стоит на земле, второй стоит у него на плечах, третий стоит на плечах у второго и т.д. Каждый атлет характеризуется силой s_i и массой m_i . Сила это максимальная масса, которую атлет способен держать у себя на плечах. Известно, что если атлет тяжелее, то он и сильнее, но атлеты равной массы могут иметь различную силу.

a) В каком порядке выстроиться спортсменам, чтобы получилась башня высоты n ?

Если это возможно, конечно. $\mathcal{O}(n \log n)$.

b) Какова максимальная высота башни? За $\mathcal{O}(n^2)$ динамикой.

c) (*) Какова максимальная высота башни? За $\mathcal{O}(n \log n)$ жадностью.

10. **Выбор заявок в маршрутке**

Маршрутка совершает рейс от первой до n -й остановки. В маршрутке m мест для пассажиров. Есть k человек, про каждого заранее известно, что он хочет доехать от остановки s_i до f_i . Проезд для пассажира стоит 1 вне зависимости от расстояния между остановками. Максимизируйте прибыль, при условии, что можно выбирать, кого сажать в маршрутку на каждой остановке. $n, m, k \leq 10^6$.

11. (*) **Идейная**

Машина тратит единицу топлива на километр, имеет бак объёма k и находится в начале прямой дороги в точке 0. Для всех $i \in \mathbb{N} \cup \{0\}$ на i -м километре дороги есть заправочная станция со своей положительной ценой c_i . Определите за время $\mathcal{O}(n)$, как проехать n километров за минимальную стоимость.

12. (*) **Подпоследовательности и сортировки**

Даны скобочные последовательности из круглых скобок. в каком порядке их склеить, чтобы получилась правильная?

13. (*) **Станки и сортировки**

Имеется n деталей и два станка. Каждая деталь должна сначала пройти обработку на первом станке, затем — на втором. При этом i -ая деталь обрабатывается на первом станке за a_i времени, а на втором — за b_i времени. Каждый станок в каждый момент времени может работать только с одной деталью. Требуется составить такой порядок подачи деталей на станки, чтобы итоговое время обработки всех деталей было бы минимальным. $\mathcal{O}(n \log n)$.

5.2 Домашнее задание

1. (2) Паросочетания

Дан взвешенный неориентированный граф из $n \leq 20$ вершин. Найдите максимальное по весу паросочетание.

2. (2) Белоснежка и гномы, которые не хотят спать

Даны n гномов. Если i -го гнома укладывать спать a_i минут, он потом спит b_i минут. Можно ли сделать так, чтобы в какой-то момент все гномы спали? $\mathcal{O}(n \log n)$.

3. (2) Интерактивная обработка заказов

В фирму поступают заказы, которые можно выполнять в произвольном порядке. В каждый момент времени можно работать ровно над одним заказом. Изначально заказов нет, i -й заказ поступает в момент времени r_i , работать над ним нужно t_i . Пусть e_i – момент окончания выполнения заказа номер i . Распределите работу над заказами так, чтобы минимизировать $\sum_i e_i$. Переходить от одного заказа к другому можно в любой момент времени (даже если заказ не доделан до конца, незаконченный заказ можно будет возобновить с того же места). Свойства заказа (r_i, t_i) не известны до момента его поступления. Всего поступит n заказов.

a) За время $\mathcal{O}(n^2)$.

b) (доп.) (+1) За время $\mathcal{O}(n \log n)$.

4. (1+1) Степени и антиклики

Будем называть *независимым множеством* или *антикликой* попарно несвязное подмножество вершин графа. Пусть в графе G есть n вершин, а максимальная степень равна d . Найдите в нём независимое множество размера хотя бы

a) $\frac{n}{d+1}$ за время $\mathcal{O}(n)$

b) $\sum_{v \in V(G)} \frac{1}{\deg(v)+1}$ за время $\mathcal{O}(n \log n)$

Считайте, что граф уже дан в памяти в виде массива, где для каждой вершины хранится список её соседей.

5. (2) Авторитеты

Есть n человек. Человек i готов примкнуть к нашей банде, если наш авторитет хотя бы a_i , при этом он к нашему авторитету прибавит b_i . Наш изначальный авторитет равен A . $a_i, b_i, A \in \mathbb{Z}$

a) Можем ли завербовать всех людей? $\mathcal{O}(n \log n)$.

b) (доп.) (+2) Какое максимальное число людей мы можем завербовать?
 $\mathcal{O}(n \log n)$. Половина балла даётся за доказательство.

6. (3) Орлы и равноправие (только группа Серёжи)

Даны n монеток, у каждой есть своя вероятность выпадения орла p_i . Нужно выбрать подмножество размера k (чётное число) из них, для которого вероятность выпадения ровно $\frac{k}{2}$ орлов при одновременном подбрасывании максимальна.

a) $\mathcal{O}(n \log n + k^3)$.

b) (доп.) (+1) $\mathcal{O}(n + k^2)$.

Не забывайте включить жадный настрой.

Не забывайте, что любую гипотезу можно проверить, вы же программисты.

Разрешается пользоваться знаниями из динамического программирования и математики.

5.3 Дополнительные задачи

7. (2+2) Окружности и отрезки

Даны n непересекающихся кругов на плоскости. Мы стоим в точке $(0, 0)$ и можем стрелять по прямой. Минимальным числом выстрелов проткнуть все круги.

- a) Все круги целиком лежат в первой четверти плоскости.
- b) Круги произвольны

8. (1+1) Школьники и ямы

n школьников упали в яму глубины S . Каждый школьник имеет рост (от ног до плеч) h_i и длину рук l_i . Школьники могут вставать друг другу на плечи, верхний школьник может вытянуть руки.

- a) Могут ли выбраться все школьники? $\mathcal{O}(n \log n)$.
- b) Какое максимальное число школьников может выбраться? $\mathcal{O}(n \log n)$.

9. Раскраской вершин графа $G = (V, E)$ называется функция $c : V \rightarrow [m]$, сопоставляющая каждой вершине G цвет от 1 до m . Раскраска называется *правильной*, если каждая пара соседних вершин имеет разные цвета.

Для неориентированного графа $G = (V, E)$ его *хроматическим числом* $\chi(G)$ называется наименьшее возможное число цветов в правильной раскраске G .

Для графа G обозначим размер его максимального полного подграфа через $\omega(G)$.

Рассмотрим на вещественной прямой замкнутые отрезки I_1, I_2, \dots, I_n . Сопоставим каждому отрезку I_i вершину v_i и каждой паре пересекающихся отрезков (I_i, I_j) ребро (v_i, v_j) . Такой граф будем называть *интервальным графом*.

Будем называть граф G *совершенным*, если для любого его индуцированного подграфа H верно $\omega(H) = \chi(H)$.

Докажите, что каждый интервальный граф совершенен. Приведите алгоритм, красящий интервальный граф $G = (V, E)$ с $|V| = n$ в $\omega(H)$ цветов за время $\mathcal{O}(n \log(n))$.

10. В фирму поступают заказы, которые можно выполнять в произвольном порядке. В каждый момент времени можно работать ровно над одним заказом. Изначально заказов нет, i -й заказ поступает в момент времени r_i , работать над ним нужно t_i .

Все заказы объединены в проекты (один заказ относится к одному проекту, заказы из одного проекта могут поступать не подряд).

Пусть e_i – момент окончания выполнения последнего (в порядке выполнения) из заказов в проекте с номером i .

Нужно распределить работу над заказами так, чтобы минимизировать $\sum_i e_i$. Свойства заказа $(r_i, t_i, \text{его проект})$ не известны до момента его поступления. Переходить от одного заказа к другому можно в любой момент времени (даже если заказ не доделан до конца).

Придумайте решение, которое не более чем в два раза хуже оптимального. Время $\mathcal{O}(n \log n)$, при условии, что всего поступит n заказов.

6 Поиск в глубину

6.1 Практика

1. **Просто ориентируй**

Ориентировать неорграф так, чтобы он стал ацикличным за $\mathcal{O}(V + E)$.

2. **Берегите деревья**

Ориентировать неорграф так, чтобы он стал сильно связным за $\mathcal{O}(V + E)$.

3. **Разве это не NP-трудно?**

Дан DAG, найти в нем гамильтонов путь за $\mathcal{O}(V + E)$.

4. **Единственный топсорт**

Дан DAG, проверить единственность топологической сортировки за $\mathcal{O}(V + E)$.

5. **dfs всемогущий**

Даны два множества вершин: A и B , за $\mathcal{O}(V + E)$ проверить, есть ли путь из какой-нибудь вершины $a \in A$ в какую-нибудь вершину $b \in B$.

6. **Чётность циклов**

За $\mathcal{O}(V + E)$ найти в неорграфе какой-нибудь цикл нечётной длины.

7. **В поисках простого цикла**

а) Найти цикл в орграфе через данное ребро за $\mathcal{O}(E)$.

б) Найти цикл в орграфе через данную вершину за $\mathcal{O}(E)$

с) Найти в неорграфе какой-нибудь цикл за $\mathcal{O}(V)$.

8. **Ценность времени**

Дано подвешенное дерево, нужно с линейным предподсчётом научиться отвечать на запрос «правда ли вершина u лежит в поддереве вершины v » online за $\mathcal{O}(1)$.

9. **Телепортация в дереве**

Дано корневое дерево и m телепортов. Для каждой вершины v дерева насчитайте самую высокую вершину, куда можно телепортироваться из поддерева v .

10. **Мосты**

Дан связный неорграф, нужно найти в нем за $\mathcal{O}(V + E)$ все мосты.

11. **dfs и веса**

В стране n аэропортов. Самолёт может сделать перелёт из аэропорта i в аэропорт j , израсходовав $w_{ij} > 0$ горючего. При этом w_{ij} может отличаться от w_{ji} , и $w_{ii} = 0$. Найдите минимальный размер бака, позволяющий добраться самолёту из любого города в любой, возможно с дозаправками.

12. **3-связность**

Проверить граф на рёберную 3-связность за $\mathcal{O}(VE)$.

13. **(*) Нельзя не пройти**

Найти в орграфе все вершины, через которые проходит любой путь $a \rightsquigarrow b$. $\mathcal{O}(V + E)$.

6.2 Домашнее задание

1. **(2) Поиск цикла**

Найти в неорграфе простой цикл через данную вершину за $\mathcal{O}(E)$.

2. **(3) Пути в дереве**

Дано дерево $T = \langle V, E \rangle$.

За $\mathcal{O}(V+E)$ вычислить для каждого ребра, сколько простых путей проходит через него.

3. **(3) Лексмин топсорт**

Найдите лексикографически минимальный из всех топологических порядков. $V, E \leq 10^6$.

4. **(3) Обмен местами**

Есть ориентированный граф. Для каждой пары вершин a, b определена функция $f(a, b)$. Вася и Петя стоят в вершинах v и p , соответственно, и хотят поменяться местами, не оказываясь ни в какой момент времени в паре вершин с $f(a, b) < d$. За какое минимальное число ходов они могут это сделать? Ход — один из них переходит в смежную вершину. $\mathcal{O}(VE)$.

Дополнительные задачи

1. **(3) Враги и двухпартийная система**

У каждой вершины не более 3 врагов. Вражда — симметричное отношение. Разбить вершины на 2 доли так, чтобы с вершиной в долю попало не более 1 врага. $\mathcal{O}(V + E)$.

2. **(4) Дополнение до сильносвязного**

Для заданного ориентированного графа G посчитать минимальное число рёбер, которые нужно добавить в граф, чтобы он стал сильно связным. Время $\mathcal{O}(V^3)$.

3. **(4) Центры дерева**

Дано дерево $T = \langle V, E \rangle$. Обозначим через $d(u, v)$ длину пути между вершинами u и v .

а) **(1)** Найдите за $\mathcal{O}(|V|)$ центр дерева — вершину c , для которой минимальна

$$\sum_{v \in V} d(v, c)$$

б) **(1)** Найдите за $\mathcal{O}(|V|^2)$ 2-центр дерева — пару вершин (a, b) , для которой минимальна

$$\sum_{v \in V} \min(d(v, a), d(v, b))$$

с) **(2)** Найдите 2-центр за $\mathcal{O}(|V|)$.

7 Кратчайшие пути

7.1 Практика

1. Кратчайший путь через достопримечательности

Дан орграф. Найти кратчайший путь, проходящий по всем k выделенным вершинам. $k \leq 10$.

2. k-bfs

Пусть в графе все ребра имеют целый вес из $[1, k]$. Придумайте, как найти кратчайшее расстояние от вершины s до остальных за $\mathcal{O}(k(V+E))$. А за $\mathcal{O}(kV + E)$?

3. Число кратчайших путей

Дан орграф и выделенная вершина s , нужно для каждой вершины v найти число кратчайших путей из v в s . $\mathcal{O}(E \log V)$.

4. Запросы к Роберту

Предподсчет за $\mathcal{O}(V^3)$ и запрос $\langle a, b, e \rangle$ за $\mathcal{O}(1)$ — существует ли кратчайший путь из a в b , проходящий через ребро e ?

5. Флойд против Беллмана

Для каждой вершины графа узнать, есть ли отрицательный цикл через эту вершину. $\mathcal{O}(V^3)$.

6. Потенциалы

Пусть дан взвешенный граф G с циклами отрицательного веса. На вершинах этого графа определим функцию $\phi(v)$ — потенциал. Заменим вес каждого ребра $w(u, v)$ на $w'(u, v) = w(u, v) + \phi(u) - \phi(v)$. Докажите, что кратчайшим путям между двумя вершинами в графе с весами w' будут однозначно соответствовать кратчайшие пути в графе с весами w .

7. Кратчайшие пути между всеми парами вершин

Пусть во взвешенном графе G нет циклов отрицательной стоимости. Докажите, что если в качестве потенциала взять кратчайшее расстояние от некоторой вершины s , то все веса w' получатся неотрицательными (если соответствующие расстояния конечны).

Покажите, как найти матрицу расстояний в графе с отрицательными весами за $\mathcal{O}(VE + V^2 \log V)$.

8. Откуда берутся графы?

Дана система из m неравенств на n переменных x_i .

Каждое неравенство имеет вид $x_i - x_j \leq \delta_{ij}$.

(а) Найти решение системы или сказать, что его не существует, за $\mathcal{O}(n \cdot m)$.

(б) Пусть все $\delta_{ij} \geq 0$, решить задачу за $\mathcal{O}(n \cdot m)$. Слишком просто? Тогда $\sum_i x_i \rightarrow \max$.

9. Расстояние — это не только сумма

Для каждой пары вершин в графе найти $w[a, b]$ — такой минимальный вес, что из a в b есть путь по рёбрам, вес которых не больше $w[a, b]$. $\mathcal{O}(V^3)$.

10. Речной граф

Даны две параллельных прямых (река). В реке есть n островов (точек). Мы хотим провести по реке корабль, представляющий собой открытый круг радиуса R , так, чтобы он не задел ни одного острова. Найти максимальный R , при котором это еще возможно, за $\mathcal{O}(n^2 \log n)$.

11. Обмен валют

Есть n валют и m обменников. i -й обменник предлагает менять валюту a_i на валюту b_i по курсу c_i/d_i . Можно ли, используя сколь угодно большие начальные сбережения и данные m обменников, сломать финансовую систему и бесконечно обогащаться? Считается, что у обменников есть бесконечное количество денег целевой валюты.

(*) А теперь тоже самое с комиссией: x_i единиц валюты a_i перейдут в $(x_i - s_i) \frac{c_i}{d_i}$ единиц валюты b_i .

12. Количество путей

Найти количество путей (необязательно простых) в графе за $\mathcal{O}(V^3 \log k)$

- а) между всеми парами вершин длины ровно k .
- б) между парой вершин длины $\leq k$.

7.2 Домашнее задание

1. Дерево Штайнера

Дан взвешенный неориентированный граф $G = \langle V, E \rangle$. Веса рёбер неотрицательны. В графе есть подмножество вершин T , которые мы назовем терминалами. Минимальное дерево Штайнера – это связный подграф графа G минимального веса, содержащий все терминалы. Требуется найти такой подграф и доказать, что он является деревом.

- (a) Доказать, что искомым подграф – дерево.
- (b) Пусть $|T| = 3$, решить за $\mathcal{O}(E \log V)$.
- (c) Пусть $|T| = 4$, решить за $\mathcal{O}(V^3)$.
- (d) (*). Что делать, если в графе есть отрицательные рёбра? является ли всё ещё минимальный по весу подграф деревом?

2. Расстояния в меняющемся графе

Нужно научиться на запрос «уменьшился вес ребра» за $\mathcal{O}(V^2)$ пересчитывать матрицу расстояний. Считайте, что в графе не было и не появилось отрицательных циклов.

Дополнительные задачи

3. bfs и длинная очередь

Постройте матрицу $n \times n$, состоящую из клеток-стенок и пустых клеток, на которой при запуске BFS из какой-то клетки максимальный размер очереди будет $\omega(n)$. Ходить можно между клетками, смежными по стороне.

4. Форд-Беллман и число итераций

Пусть на вершинах графа задан порядок: v_1, v_2, \dots, v_n . Пусть алгоритм Беллмана-Форда на каждой стадии рассматривает рёбра в таком порядке: сначала рёбра, ведущие из меньшей вершины в большую (в порядке возрастания номера исходящей вершины), а потом рёбра, ведущие из большей вершины в меньшую (в порядке убывания номера исходящей вершины). Докажите, что если в графе нет циклов отрицательного веса, то алгоритм найдет все кратчайшие пути из v_1 за $\frac{n}{2}$ итераций.

5. Форд-Беллман и число итераций 2

Продолжение задачи про улучшения Беллмана-Форда. Пронумеруем вершины в порядке случайной перестановки. Покажите, что матожидание числа итераций Беллмана-Форда есть $\frac{n}{3}$.

6. Форд-Беллман и число итераций 3

Покажите, что на случайном графе ($a_i \in [1, n]$, $b_i \in [1, n]$, $w_i \in [1, C]$, все числа целые, генерируются равномерным распределением, C любая фиксирована один раз заранее) Форд-Беллману хватит $\mathcal{O}(\log V)$ итераций.

7. Почему 5?

Дан неориентированный граф G . Посчитайте количество простых циклов длины 5. $\mathcal{O}(V^3)$.

8. Почему без лог?

Дан орграф, посчитайте от начальной вершины до всех остальных кратчайшие расстояния относительно метрики «вес самого легкого ребра на пути». $\mathcal{O}(V + E)$.

9. Максимально отрицательный цикл

Дан взвешенный орграф с положительными целыми весами, не превосходящими C . Найдите в нем цикл минимального среднего веса за $\mathcal{O}(VE \log C)$.

10. Сумма двух

Есть взвешенный неограф. Найти путь из s в t такой, что сумма двух максимальных ребер на пути минимальна. $\mathcal{O}((V + E) \cdot \text{poly}(\log))$.

11. Модульный граф

Пусть длина пути определяется как сумма весов всех ребер по модулю n . Найти кратчайший путь за $O((V + E) \cdot n)$.

12. Рассмотрим граф, на каждом ребре которого написан нолик или единица. Каждому пути в этом графе соответствует некоторая бинарная строка. Строки сравниваются как двоичные числа (старшие разряды слева), среди одинаковых чисел меньше то, у которого меньше ведущих нулей. Для задачи нахождения кратчайших путей относительно такой метрики докажите корректность или приведите контрпример для алгоритма

- а) Флойда
- б) Дейкстры
- в) Беллмана-Форда

13. Для обычной функции расстояния алгоритм Беллмана-Форда при работе делает несколько стадий обновления всех ребер и находит расстояния от вершины A до всех остальных за $V - 1$ итерацию. Предположим, что на каждой стадии алгоритм обновляет рёбра согласно некоторому фиксированному порядку на множестве E , общему для всех стадий. Зададим новую «длину пути»: определим её равной минимальному весу ребра на этом пути. При инициализации положим расстояние от A до A равным бесконечности, а до остальных вершин неопределённым, релаксация из вершин с неопределённым расстоянием не проводится. Будет ли работать алгоритм Беллмана-Форда с так определенной длиной? Сколько итераций он сделает в худшем случае для направленного графа?

8 Остовные деревья

8.1 Практика

1. Аэропутешествия

В стране n аэропортов. Самолет может сделать перелет из аэропорта i в аэропорт j , израсходовав w_{ij} горючего. При этом $w_{ij} = w_{ji}$, и $w_{ii} = 0$. Требуется найти минимальный размер бака, позволяющий добраться самолету из любого города в любой, возможно с дозаправками. Решить за $\mathcal{O}(n^2)$.

2. Приближение коммивояжера

Найдите 2-приближение для задачи коммивояжера в графе с весами, удовлетворяющими неравенству треугольника.

3. (*) СНМ для ленивых

Докажите, что СНМ только с эвристикой «сжатие путей» работает за амортизированный $\mathcal{O}(\log n)$ на запрос.

Подсказка #1: используйте знания про лёгкие и тяжёлые рёбра.

Подсказка #2: посмотрите как меняются размеры поддеревьев.

4. Казалось бы, причём здесь СНМ?

У нас есть массив длины n , мы хотим выполнить m запросов вида «покрасить отрезок $[l_i, r_i]$ массива в цвет c_i » и вывести, что получилось в конце. Решите за $\mathcal{O}(n \log m)$.

Подсказка: попытайтесь выполнять запросы, начиная с последнего.

5. Алгоритм Борувки

Рассмотрим следующий алгоритм поиска минимального покрывающего дерева:

while (в графе больше одной вершины):

- Для каждой вершины найдем самое легкое инцидентное ей ребро и добавим его в множество S (одно и то же ребро может быть выбрано дважды).
- Добавим все ребра из множества S в ответ.
- Стянем граф по ребрам из S .

Докажите, что такой алгоритм найдет минимальное покрывающее дерево в случае, если веса всех ребер в графе различны, при этом время работы будет $\mathcal{O}(E \log V)$.

Придумайте, как модифицировать алгоритм, если возможны равные веса.

Ценность алгоритма по сравнению с Краскалом и Примом в том, что часто получается $\mathcal{O}(E)$.

Постройте пример для алгоритма Борувки, на котором он делает $\Theta(1)$ фаз.

Занимательный факт. Борувка в худшем случае работает за $\mathcal{O}(E \log_{E/V} V)$.

Обычные Прим и Краскал дольше. Прим с d -ичной кучей внутри столько же.

6. Минимальный цикл

Найти во взвешенном неорграфе такой цикл, что

максимальный вес ребра этого цикла минимален. $\mathcal{O}((V + E) \log E)$.

7. MST в меняющемся графе

Дан взвешенный связный неориентированный граф $G = \langle V, E \rangle$ и некоторое минимальное остовное дерево на нём. Пусть некоторого ребра $e \in E$ изменился вес. По графу, остовному дереву, ребру e и его новому весу найдите новое минимальное остовное дерево за $\mathcal{O}(V + E)$.

8. Единственность MST

- а) Пусть все ребра графа имеют различный вес. Докажите, что минимальное покрывающее дерево единственно.
- б) Проверить, что минимальное по весу остовное дерево единственно. $\mathcal{O}(E \log V)$.

9. Два различных MST

Пусть дан взвешенный неорграф с неединственным минимальным остовным деревом. Найдите два различных минимальных остова графа. $\mathcal{O}(E \log V)$.

10. Второе по минимальности MST

Найдите за полиномиальное время второе по весу остовное дерево в неорграфе.

11. (*) Все деревья связаны

Пусть дан взвешенный связный неорграф $G = \langle V, E \rangle$ с выделенной вершиной s . Все веса положительны и различны. Могут ли какое-либо минимальное покрывающее дерево в G и какое-либо дерево кратчайших путей из s не иметь ни одного общего ребра? Если да, приведите пример. Если нет, докажите, что такого не может быть.

8.2 Домашнее задание

1. Второе по минимальности MST

Найдите за $O(V^2 + E)$ второе по весу остовное дерево в неографе.

Указание: улучшите версию решения с практики с попыткой добавить новое ребро в дерево. Мы ещё не знаем заклинания «LCA!» пользоваться им нельзя. Есть простое решение с использованием только известной вам теории.

2. Краскал наоборот

Пусть дан связный взвешенный неорграф, будем рассматривать его ребра в порядке невозрастания веса и удалять текущее ребро, если связность графа при этом не нарушается. Докажите, что этот алгоритм находит минимальный остов, или придумайте контрпример.

3. Насколько сжатие путей быстрое?

Доказать, что СНМ с одной эвристикой сжатия пути, к которому поступают следующие запросы «сперва только join-ы вида $p[\text{root}] = x$ (где root – именно корень, а x – любая вершина другого множества), затем только get-ы», работает за $O(n + m)$, где n – число элементов, m – суммарное число запросов.

4. Сжатие путей и максимум на пути в дереве

Дано дерево из n вершин, с весами на рёбрах. Даны m вертикальных путей (путь $a \rightarrow b$ вертикальный, если b – предок a). Посчитайте на каждом вертикальном пути максимум весов рёбер. $n, m \leq 10^6$.

Указание #1: считайте почти «в лоб», но используйте сжатие путей (похоже на задачу с практики).

Указание #2: важно пользоваться тем, что задача offline (вам сразу даны все пути), а не поступают по одному (online).

Дополнительные задачи

1. Сумма трёх рёбер

Есть взвешенный неорграф. Найти путь из s в t такой, что сумма трёх максимальных рёбер на пути минимальна. $O((n + m) \cdot \text{poly}(\log))$.

2. Ориентированное остовное дерево

Дан взвешенный оргграф, постройте ориентированное к корню остовное дерево с корнем в вершине 1 минимального веса. За полином.

3. Второе остовное дерево lvl. 2

Дан взвешенный связный неорграф G , вес его минимального остова равен w . Найдите за полиномиальное время минимальный по весу среди таких остовов G , вес которых строго превосходит w .

4. Дан взвешенный граф с положительными весами, в вершинах его стоят числа. Требуется доставить все числа в вершину 0. За то, чтобы провести число a по ребру веса w надо заплатить сумму aw . Если два числа находятся в одной вершине, то их можно слить, заменив на максимум. Постройте алгоритм доставки, дающий константное приближение к оптимальному ответу за полиномиальное время.

9 Потоки и разрезы

9.1 Практика

1. Дан двудольный граф. Каждой вершине сопоставлено число a_v .
 - (a) Выберите максимальное количество рёбер так, чтобы степени вершин были не более 1.
 - (b) Выберите максимальное количество рёбер так, чтобы степени вершин были не более a_v .
2. Даны девочки, мальчики и собачки. Для каждой пары “мальчик, девочка” известно, хочет ли девочка дружить с мальчиком. Для каждой пары “собачка, девочка” известно, нравится ли собачка девочке. Нужно максимальному количеству девочек выделить по мальчику и собачке так, что:
 - Каждый мальчик не более чем с одной девочкой.
 - Каждая собачка не более чем у одной девочки.
 - Тройки гармоничны: девочка и хочет дружить с выбранным ей мальчиком, и собачка ей нравится.
3. Дан неориентированный граф. Необходимо ориентировать его так, чтобы максимальная исходящая степень была минимальна.
 - (a) $\mathcal{O}(E^2 \log V)$
 - (b) $\mathcal{O}(E^2)$
4. По правилам футбольного турнира в каждом матче должна победить одна из команд, то есть, не бывает ‘ничьих’. Вам дана матрица уже сыгранных матчей. Можно ли так доиграть турнир, чтобы каждая команда выиграла заданное число раз? Каждая команда играет с каждой, для каждой команды известно, сколько игр она выиграла.
5. Рассмотрим ориентированный граф. За одно действие можно удалить все входящие в вершину i рёбра за стоимость $a_i \geq 0$, или все исходящие из вершины i рёбра за стоимость $b_i \geq 0$. Необходимо удалить все рёбра графа за минимальную стоимость.
6. Каждой вершине ориентированного графа сопоставлено число (не обязательно положительное) — её вес. Найдите замкнутое подмножество вершин максимальной суммарной стоимости. Подмножество вершин называется замкнутым, если из него не исходят рёбра в другую часть графа.
7. Есть заказы и инструменты. Для каждого заказа известен список инструментов, который нужен, чтобы его выполнить. Каждый инструмент сделан умелыми японскими рабочими, поэтому бесконечно прочный, его можно один раз купить и много раз использовать. У каждого инструмента есть цена p_i . У каждого заказа есть прибыль, которую можно получить, выполнив заказ. Вы — бедный китайский рабочий. У вас изначально нет инструментов, но зато вы можете под нулевой процент в банке взять сколь угодно большой кредит, чтобы купить инструментов.
 - (a) Вопрос: какую максимальную прибыль вы можете получить?
 - (b) А теперь тот же вопрос, но ещё есть разные скидочные предложения!
Скидка позволяет два инструмента i, j купить по специальной цене d : $\max(p_i, p_j) < d < p_i + p_j$. Каждый инструмент присутствует не более чем в одном скидочном предложении.
8. Есть ориентированный граф с начальной и конечной вершинами. В начальной вершине есть K грузовиков. Грузовикам нужно попасть в конечную вершину. Время дискретно. За единицу времени каждый грузовик или стоит на месте, или перемещается в одну из соседних вершин. В любой вершине могут одновременно стоять несколько грузовиков. По любому из рёбер в каждый момент времени должен ехать не более чем один грузовик. Минимизируйте время, когда все грузовики окажутся в конечной вершине.
 - (a) $\mathcal{O}(\text{poly}(V, E, K))$
 - (b) $\mathcal{O}(K(V + K)E)$

9.2 Домашнее задание

1. Разбейте вершины ориентированного графа на циклы. Т.е. каждая вершина должна быть покрыта ровно одним циклом. Либо скажите, что это невозможно. $\mathcal{O}(EV)$.
2. Дан граф и выделенные вершины s, t . Нужно проверить, правда ли существует единственный минимальный s - t разрез.
 - (a) $\mathcal{O}(\text{poly}(V, E))$
 - (b) $\mathcal{O}(E)$ при условии, что нам уже известен максимальный поток (сам поток, не только его величина).
3. В неориентированном графе без кратных рёбер необходимо удалить минимальное число рёбер так, чтобы увеличилось количество компонент связности.
 - (a) $\mathcal{O}(V \cdot \text{Flow})$, где Flow — время работы какого-нибудь алгоритма поиска максимального потока.
 - (b) $\mathcal{O}(E^2)$.
4. *(только группа Серёжи)* Есть ориентированный граф с начальной и конечной вершинами. В начальной вершине есть K грузовиков. Грузовикам нужно попасть в конечную вершину. Время дискретно. За единицу времени каждый грузовик или стоит на месте, или перемещается в одну из соседних вершин. В любой вершине могут одновременно стоять несколько грузовиков. По любому из рёбер в каждый момент времени должен ехать не более чем один грузовик. Минимизируйте время, когда все грузовики окажутся в конечной вершине.
 - (a) $\mathcal{O}(\text{poly}(V, E, K))$
 - (b) $\mathcal{O}(K(V + K)E)$
5. *(только группа Влада)* Есть заказы и инструменты. Для каждого заказа известен список инструментов, который нужен, чтобы его выполнить. Каждый инструмент сделан умелыми японскими рабочими, поэтому бесконечно прочный, его можно один раз купить и много раз использовать. У каждого инструмента есть цена p_i . У каждого заказа есть прибыль, которую можно получить, выполнив заказ. Вы — бедный китайский рабочий. У вас изначально нет инструментов, но зато вы можете под нулевой процент в банке взять сколь угодно большой кредит, чтобы купить инструментов.
 - (a) Вопрос: какую максимальную прибыль вы можете получить?
 - (b) А теперь тот же вопрос, но ещё есть разные скидочные предложения!
Скидка позволяет два инструмента i, j купить по специальной цене d : $\max(p_i, p_j) < d < p_i + p_j$. Каждый инструмент присутствует не более чем в одном скидочном предложении.

Подсказка: возможно, поможет одна из задач с практики.

Дополнительные задачи

1. Пусть алгоритм Форда-Фалкерсона ищет дополняющие пути DFS-ом с фиксированным порядком просмотра рёбер в каждой вершине, и пропускает максимум по найденному дополняющему пути. Постройте вместе с порядком ребер (или докажите, что такого не существует) ориентированный граф:
 - (a) с целочисленными пропускными способностями, на котором алгоритм работает за экспоненту от V (пропускные способности могут быть большими, но должны быть полиномиальной суммарной длины).
 - (b) с вещественными пропускными способностями, на котором алгоритм не завершается за конечное количество шагов.
 - (c) с вещественными пропускными способностями, на котором алгоритм не сходится к корректному значению величины максимального потока.

2. Дана укладка неориентированного планарного графа: вершинам сопоставлены точки на плоскости, рёбрам — непересекающиеся отрезки между вершинами. У рёбер есть пропускные способности. Даны две вершины s и t , лежащие на одной грани. Требуется за $\mathcal{O}(\text{Dijkstra})$ найти величину максимального потока из s в t .
3. Какое максимальное количество уголков можно разместить на шахматной доске $n \times m$ с дырками? Уголком называется фигура, состоящая из трех клеток: центральная клетка черного цвета и две соседних с ней белых клетки со смежными сторонам.

10 Бинарные деревья поиска

10.1 Практика

1. `lower_bound`

Как реализовать `next(x)`, `lower_bound(x)` в BST за $\mathcal{O}(h)$?

2. Объединение AVL

Пусть вам даны два AVL-деревя T_1 и T_2 . Придумайте, как построить AVL-дерево T , являющееся объединением деревьев T_1 и T_2 за время:

(a) $\mathcal{O}(\text{height}(T_1) \times \text{size}(T_2))$, если $\text{size}(T_1) \geq \text{size}(T_2)$.

(b) $\mathcal{O}(\text{size}(T_1) + \text{size}(T_2))$

3. `k-min`

Как вывести k минимальных элементов в AVL-дереве за $\mathcal{O}(k)$?

А для произвольного BST?

4. Минимум на отрезке

Улучшите BST, хранящее пары $\langle x_i, y_i \rangle$: научите его находить $\min y_i : L \leq x_i \leq R$ за $\mathcal{O}(h)$.

5. Операции над AVL

a) Удаление за $\mathcal{O}(\log n)$.

b) Пусть `root.l.h = root.r.h + 3`. Как перебалансировать дерево за $\mathcal{O}(1)$?

c) Пусть `root.l.h = root.r.h + k`. Как перебалансировать дерево за $\mathcal{O}(k)$?

d) Merge за $\mathcal{O}(\log n)$ (два способа).

e) Split за $\mathcal{O}(\log^2 n)$. (*) за $\mathcal{O}(\log n)$.

f) (*) Уменьшите количество дополнительной информации до двух бит на вершину.

6. Быстрый отсортированный массив Нужно быстро выполнять запросы:

- `get_ith_element(i)` (i -й по величине элемент);
- `get_position(x)` (номер элемента в отсортированном массиве);
- `add(i, x, y)` (вставить x на i -ю позицию; гарантируется, что $S[i-1] < x < S[i]$);
- `del(i)` (удалить элемент на i -й позиции).

7. Быстрый массив Нужно быстро выполнять запросы: `get(i)`, `set(i, y)`, `insert(i, y)` (вставить x после i -го элемента), `del(i)`, `rotate(k)` (циклический сдвиг на k элементов).

8. Модификации отрезка

a) Запросы: `insert(i, x)`, `del(i)`, `get_sum(l, r)`, `set(l, r, value)`.

b) Добавим запросы `reverse(l, r)` и `rotate(k)`.

9. Покажите, что любые два корректные дерева поиска, построенные на одном и том же множестве ключей, можно получить друг из друга последовательностью поворотов.

10.2 Домашнее задание

1. Покажите, что:

(a) Добавление в AVL-дерево требует $\mathcal{O}(1)$ вращений.

(b) Удаление из AVL-деревя может потребовать $\Omega(\log n)$ вращений.

2. Точка в стакане

Дано множество точек на плоскости. Нужно быстро обрабатывать запросы:

- добавить/удалить точку,
- вывести любую точку внутри области $d_i \leq y \leq u_i, x \leq r_i$.

3. Рассмотрим обычное несбалансированное BST. Если делать добавление в лоб спуском вниз, время работы на один запрос добавления может быть $\Omega(n)$. В процессе мы можем поддерживать для каждой вершины ссылки на детей и отца, а также глубину вершины (расстояние до корня). Научитесь делать ту же самую работу – поддерживать для каждой вершины детей, отца, глубину быстрее, а именно за $\mathcal{O}(\log n)$ на один запрос добавления.
4. Запросы online за $\mathcal{O}(\log n)$:
 - добавить пару $\langle x, y \rangle$
 - посчитать сумму y по всем таким парам, что $l \leq x \leq r$
 - посчитать сумму x по всем таким парам, что $l \leq y \leq r$
5. Придумайте структуру данных, поддерживающую упорядоченный по значению список S целых чисел, которая умеет отвечать на запросы:
 - `insert(x)` — вставить x в S , если его там не было.
 - `delete(x)` — удалить x из S , если он там был.
 - `S[k]` — вернуть k -тый по порядку элемент из S (то есть это k -я порядковая статистика множества).
 - `pos(x)` — вернуть k такое, что $S[k] = x$ (или \emptyset , если такого нет).

Каждый запрос должен обрабатываться за $\mathcal{O}(\log |S|)$.

10.3 Дополнительные задачи

1. Дано корневое дерево, на вершинах которого могут быть пометки. Запросы: пометить вершину, снять пометку с вершины, число помеченных вершин в поддереве. Предобработка за $\mathcal{O}(n)$, online-запросы за $\mathcal{O}(\log n)$.
2. Дан невыпуклый несамопересекающийся многоугольник из n вершин.
 - (a) Даны m точек. За $\mathcal{O}((m + n) \log n)$ для каждой точки определить, внутри она или снаружи.
 - (b) Используя предподсчёт за $\mathcal{O}(n \log n)$ научиться в online по точке за $\mathcal{O}(\log n)$ определять, внутри она или снаружи.
3. Постройте тест, на котором недо-AVL-дерево, которое делает только малые вращения, делает $\omega(n \log n)$ операций после n запросов.
 Формально: изначально дерево пусто, нужно n раз вызвать `add(root, x_i)` для некоторой последовательности x_i , что суммарное время работы $\omega(n \log n)$.
 Либо докажите, что такого теста нет.