Практика по алгоритмам, ВШЭ

Владислав Кораблинов, Антон Гардер*

Осень, 2020

*Составители сборника не всегда являются авторами задач. Авторы не указаны в учебных целях.

1 Практика 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 $\mathbb{N} \to \mathbb{N}$ или $\mathbb{N} \to \mathbb{R}_{>0}$ (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 $\mathbb{N} \to \mathbb{N}$ 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 $a_1, a_2, \cdots, a_n \in \mathbb{N}$ 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 $a_1, a_2, \cdots, a_n \in \mathbb{N}$. 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 $a_1, a_2, \cdots, a_n \in \mathbb{N}$. 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

1. Дайте ответ для двух случаев $\mathbb{N} \to \mathbb{N}$ и $\mathbb{N} \to \mathbb{R}_{>0}$:

   (a) Если в определении $\mathcal{O}$ опустить условие про N (т.е. оставить просто $\forall n$), будет ли полученное определение эквивалентно исходному?

   (b) Тот же вопрос про o.

2. Считайте здесь, что $\forall n : f(n) > 1 \wedge g(n) > 1$. Правда ли, что $f(n) = o(g(n)) \Rightarrow 2^{f(n)} = o(2^{g(n)})$?

3. 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.

4. Дана последовательность $a_1, a_2, \cdots, a_n \in \mathbb{N}$ и $S \in \mathbb{N}$. Найти l, r $(1 \le l \le r \le n)$ 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 $\sum_{j=i}^{r_i} a_j \le S$ за $\mathcal{O}(n)$ для каждого i.
   - Найдите за $\mathcal{O}(n)$ ответ задачи, если известны $r_1, \ldots, r_n$.
   - Докажите, что $r_i \le r_{i+1}$
   - Пользуясь предыдущим пунктом найдите все r_i за $\mathcal{O}(n)$.

5. Дана последовательность $a_1, a_2, \cdots, a_n \in \mathbb{N}$.

   (a) За $\mathcal{O}(n)$ для каждого a_i найти самый правый из элементов, которые левее и меньше его.

   (b) За $\mathcal{O}(n)$ для каждого a_i найти самый левый из элементов, которые правее и меньше его.

   (c) За $\mathcal{O}(n)$ найти l, r $(1 \le l \le r \le n)$ такие, что значение $(r - l + 1)\min_{i \in [l,r]} a_i$ было бы максимально.

   (d) За $\mathcal{O}(n)$ найти l, r $(1 \le l \le r \le n)$ такие, что значение $\left(\sum_{i \in [l,r]} a_i\right) \min_{i \in [l,r]} a_i$ было бы максимально.

6. 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.

7. *(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

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

2. Определить асимптотику (считайте, что при $n \le 100$ 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 $a_1, a_2, \cdots, a_n \in \mathbb{Z}$. Найти l, r $(1 \le l \le r \le n)$ 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 $\sigma =< a_1, a_2, \cdots, a_m >$, где каждый $a_i \in [n] = \{1, 2, \cdots, n\}$. 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 $y \ne x, f_\sigma[y] \equiv 0 \mod 2$. Требуется найти x за один проход по последовательности, используя $\mathcal{O}(\log n + \log m)$ бит памяти.

12. Дана последовательность $\sigma =< a_1, a_2, \cdots, a_m >$, 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 длинну 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 сортированном 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 $o(n \log n)$ 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 $a[\ldots]$ 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 $x_1, \ldots, x_m$ 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 сортированных 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 сортированном 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 a).

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 $m \le n$ различных элементов.

    (a) Пусть зафиксирован набор частот элементов $p_i > 0, i = 1 \ldots m$. 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: $n \ln n \ge \ln n! = n \ln n - n + \mathcal{O}(\ln n)$.

    (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 — $\mathcal{O}(n \log n)$.

3. (a) Дано множество из n точек на плоскости. Найти пару ближайших точек за $\mathcal{O}(n \log n)$.

   (b) Дано множество из n векторов на плоскости. Разрешается координату любого вектора умножить на -1. Найти пару векторов, чья сумма минимальна, за $\mathcal{O}(n \log n)$.

4. 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:

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

   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.

   (a) Докажите, что для любого скошенного влево дерева $|T| \geq 2^{\mathbf{rank}(T)} - 1$ ($|T|$ обозначает количество вершин в дереве T).

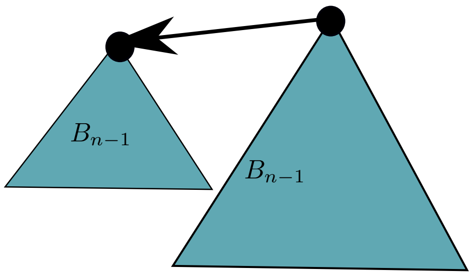
   (b) Придумайте, как слить две скошенные влево кучи H_1 и H_2 за время $\mathcal{O}(\log |H_1| + \log |H_2|)$.

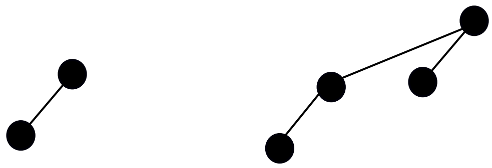
   (c) Придумайте, как используя операцию слияния, построенную на предыдущем шаге, реализовать операции:

      - `Insert(x)` — добавление элемента x в кучу,
      - `ExtractMin()` — удаление минимального элемента из кучи.

5. 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}.





B_1 B_2

(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 $\overline{a_k a_{k-1} \cdots a_2 a_1}$ 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: $\overline{a_k a_{k-1} \cdots a_2 a_1}$

- 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 $1, 2, 3, 4, 5$:

Рис. 1: Лист [a] (число: 1)

Рис. 2: Лист [a b] (число: 2)

Рис. 3: Лист [a b c] (число: 10)

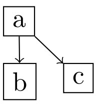


Рис. 4: Лист [a b c d] (число: 11)

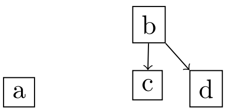
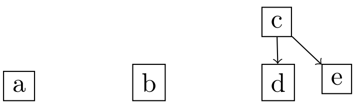


Рис. 5: Лист [a b c d e] (число: 12)



Придумайте, как реализовать следующие операции со списком длины n:

(a) Добавление элемента в начало списка за $\mathcal{O}(1)$.

(b) Доступ к i-му элементу за $\mathcal{O}(\log n)$.

(c) 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. Стуктура 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. (a) Приведите вероятностный алгоритм поиска медианы в массиве различных чисел со средним временем работы $\mathcal{O}(n)$

   (b) 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 $p_1, p_2, \ldots p_m$, 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.
   $$d(x, \texttt{median}) = |\texttt{pos}(x) - \texttt{pos}(\texttt{median})|,$$
   где $\texttt{pos}(x)$ — позиция элемента x в отсортированном массиве.
   (b) По значению.
   $$d(x, \texttt{median}) = |x - \texttt{median}|.$$
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 $\frac{\sum_{i=1}^{k} a_{p_i}}{\sum_{i=1}^{k} b_{p_i}} \to \max$. 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 $1 \ldots d$. 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 $f(n)$.