АиСД у2023. Первый семестр

Домашние задания М3134-М3137

⟨Версия от 25 декабря 2023 г.⟩

Темы

1 Время работы алгоритма. О-нотация **1**
1.1 Устная часть . . . 2

2 Квадратичные сортировки. Сортировка слиянием. Два указателя **3**
2.1 Устная часть . . . 4

3 Двоичная куча. Сортировка кучей **5**
3.1 Устная часть . . . 6

4 Сборная солянка **7**
4.1 Устная часть . . . 8

5 Бинарный и тернарный поиск **9**
5.1 Устная часть . . . 10

6 Стек. Очередь. Амортизационный анализ **11**
6.1 Устная часть . . . 12

7 Связные списки. Pointer Machine. **13**
7.1 Устная часть . . . 14

8 Система непересекающихся множеств **15**
8.1 Устная часть . . . 16

9 Быстрая сортировка. K-я порядковая статистика **17**
9.1 Устная часть . . . 18
9.1.1 Алгоритмы . . . 18
9.1.2 Математика . . . 18

10 Динамическое программирование — 1 **19**
10.1 Устная часть . . . 20

11 Динамическое программирование — 2 **21**
11.1 Устная часть . . . 22

12 Разные задачи на динамическое программирование **23**
12.1 Устная часть . . . 24

Неделя 1. Время работы алгоритма. О-нотация

Устная часть

1.1. Докажите по определению, что если $f_1(n) = \mathcal{O}(g_1(n))$ и $f_2(n) = \mathcal{O}(g_2(n))$, то $f_1(n) + f_2(n) = \mathcal{O}(g_1(n) + g_2(n))$, если $f_i(n)$ и $g_i(n)$ неотрицательны.

1.2. Докажите по определению, что $\max(f(n), g(n)) = \Theta(f(n) + g(n))$, если $f(n)$ и $g(n)$ неотрицательны.

1.3. Докажите по определению, что $\sum\limits_{i=1}^{n+5} 2^i = \mathcal{O}(2^n)$.

1.4. Докажите по определению, что $\frac{n^3}{6} - 7n^2 = \Omega(n^3)$.

1.5. Докажите по определению, что $2^n \cdot n^2 = \mathcal{O}(2.1^n)$.

1.6. Докажите по определению, что $\log_3(n^2 + 5n + 3) = \Theta(\log n)$.

1.7. Докажите или опровергните следующие утверждения. Для опровержения достаточно привести пример, на котором утверждение неверно. Все функции положительные.

(a) Если $f(n) = \mathcal{O}(g(n))$, то $g(n) = \mathcal{O}(f(n))$

(b) Если $f(n) = \mathcal{O}(g(n))$, то $2^{f(n)} = \mathcal{O}(2^{g(n)})$

(c) $f(n) = \mathcal{O}(f(n)^2)$

(d) $f(n) = \Theta(f(\frac{n}{2}))$

(e) Если $f(n) = \mathcal{O}(g(n))$, то $g(n) = \Omega(f(n))$

1.8. Пусть $p(n) = \sum\limits_{i=0}^{d} a_i n^i$, где $a_d > 0$ — полином степени d от n. Используя определения, полагая, что k — константа, докажите, что:

(a) Если $k \geq d$, то $p(n) = \mathcal{O}(n^k)$

(b) Если $k = d$, то $p(n) = \Theta(n^k)$

(c) Если $k \leq d$, то $p(n) = \Omega(n^k)$

1.9. Придумайте две положительные функции $f(n)$ и $g(n)$, такие что не выполнено ни одно из двух: $f(n) = \mathcal{O}(g(n))$ и $f(n) = \Omega(g(n))$.

1.10. Докажем, что $n^2 = \mathcal{O}(n)$ по индукции. Для этого докажем, что для любого k верно, что $kn = \mathcal{O}(n)$. Действительно, для $k = 1$ это верно. Пусть утверждение верно для $k-1$, тогда $kn = (k-1)n + n = \mathcal{O}(n)$. Таким образом, это верно для всех k, а значит и для $k = n$, таким образом $n \cdot n = \mathcal{O}(n)$. Где ошибка в этом доказательстве?

1.11. Докажите или опровергните, что $\log(n!) = \Theta(n \log n)$.

1.12. Докажите или опровергните, что $\sum\limits_{i=1}^{n} \frac{n}{i} = \mathcal{O}(n \log n)$.

1.13. Докажите по индукции, что если $T(n) = 2T(\sqrt{n}) + 1$, то $T(n) = \mathcal{O}(\log n)$. В этом и следующих заданиях в качестве базы индукции можно полагать, что $T(1) = 1$.

1.14. Докажите по индукции, что если $T(n) = 3T\left(\frac{n}{2}\right) + 1$, то $T(n) = \Omega(n)$.

1.15. Докажите по индукции, что если $T(n) = \log n \cdot T\left(\frac{n}{\log n}\right) + n$, то $T(n) = \mathcal{O}(n \log n)$.

1.16. Докажите по индукции, что если $T(n) = 2T\left(\frac{n}{2} + \log n\right) + n$, то $T(n) = \mathcal{O}(n \log n)$.

1.17. Докажите по индукции, что если $T(n) = 2T\left(\frac{n}{2} + 20\right) + n$, то $T(n) = \mathcal{O}(n \log n)$.

1.18. Найдите Θ-асимптотику времени работы алгоритма, построив дерево рекурсии, если $T(n) = T\left(\frac{n}{3}\right) + \log_2(n)$.

1.19. Доказательство Мастер-теоремы. Пусть $T(n) = aT\left(\frac{n}{b}\right) + n^c$. Докажите, что:

(a) Если $c < \log_b a$, то $T(n) = \mathcal{O}(n^{\log_b a})$

(b) Если $c > \log_b a$, то $T(n) = \mathcal{O}(n^c)$

(c) Если $c = \log_b a$, то $T(n) = \mathcal{O}(n^c \log n)$

Подсказка. Для начала оцените количество запусков на каждом уровне рекурсии, а также размер задачи (то есть, n) для каждого такого запуска. После этого оцените суммарное количество совершенных действий на каждом уровне рекурсии и просуммируйте. Полученная сумма в каждом из трех случаев должна красиво свернуться.

1.20. Пусть время работы алгоритма A равно $T_A(n) = 7T_A\left(\frac{n}{2}\right) + n^2$, а время работы алгоритма B равно $T_B(n) = aT_B\left(\frac{n}{4}\right) + n$. При каких значениях a второй алгоритм работает асимптотически быстрее первого?

Неделя 2. Квадратичные сортировки. Сортировка слиянием. Два указателя

Устная часть

2.21. Даны k отсортированных массивов, сумма длин которых равна n. Придумайте алгоритм, позволяющий объединить все эти массивы в один отсортированный массив. Время работы $\mathcal{O}(nk)$.

2.22. Даны два отсортированных массива a и b длины n. Найдите такие i и j, что значение $|a_i - b_j|$ минимально. Время работы $\mathcal{O}(n)$.

2.23. Даны два отсортированных массива a и b длины n, а также число S. Найдите такие i и j, что $a_i + b_j = S$. Время работы $\mathcal{O}(n)$.

2.24. Даны два отсортированных массива a и b длины n. Найдите количество пар (i, j), таких что $a_i = b_j$. Время работы $\mathcal{O}(n)$.

2.25. Дан массив a длины n, состоящий из неотрицательных целых чисел. Найдите в данном массиве подотрезок $[l, r]$, такой что $a_l + a_{l+1} + \ldots + a_r = k$ либо определите, что его не существует. Время работы $\mathcal{O}(n)$.

2.26. Решите предыдущую задачу при условии, что нужно найти подотрезок максимальной длины.

2.27. Дан массив a длины n, состоящий из неотрицательных целых чисел. Найдите количество подотрезков массива, сумма которых равна k.

2.28. Дан массив a. Назовем *инверсией* пару индексов (i, j), такую что $i < j$ и $a_i > a_j$. Пусть в массиве длины n ровно k инверсий. Докажите, что сортировка вставками работает за $\mathcal{O}(n + k)$.

2.29. Дан массив a длины n. Найдите количество инверсий в массиве за $\mathcal{O}(n \log n)$.

Подсказка. Модифицируйте сортировку слиянием.

2.30. Придумайте способ сделать сортировку слиянием «снизу вверх», без использования рекурсии.

2.31. В отсортированный массив длины n в произвольное место вставили новый элемент. Придумайте алгоритм, позволяющий отсортировать получившийся массив за $\mathcal{O}(n)$.

2.32. Постройте для произвольного n перестановку, для которой сортировка вставками сделает наибольшее количество операций `swap`. Сколько в точности обменов совершится? Единственна ли построенная перестановка?

2.33. Дан массив a длины n. Необходимо для каждого элемента найти количество элементов, меньших его. Время работы $\mathcal{O}(n \log n)$.

2.34. Дан массив a длины n, состоящий из натуральных чисел. Требуется найти минимальное натуральное число, которого нет в массиве, за $\mathcal{O}(n \log n)$.

2.35. Даны два массива a и b длины n. Найдите такую пару (i, j), что $a_i < a_j$ и $b_i < b_j$, либо определите, что такой пары не существует. Время работы $\mathcal{O}(n \log n)$.

2.36. Сортировка называется *стабильной*, если она сохраняет порядок элементов, не различимых при помощи операции сравнения. Например, если мы хотим отсортировать массив пар чисел, а при сортировке мы сравниваем пары только по первым элементам, если сортировка является стабильной, массив $[(1,3),(0,4),(1,0),(0,5)]$ будет отсортирован следующим образом: $[(0,4),(0,5),(1,3),(1,0)]$. Проанализируйте сортировки пузырьком, выбором, вставками и слиянием, и выясните, являются ли они стабильными.

2.37. Докажите, что любую сортировку можно сделать стабильной, потратив $\mathcal{O}(n)$ дополнительной памяти, не меняя время работы сортировки.

2.38. Дан массив a длины n. Постройте массив b длины n, состоящий из чисел от 0 до $n-1$, чтобы было выполнено следующее свойство: если $a_i < a_j$, то $b_i < b_j$. Время работы $\mathcal{O}(n \log n)$.

Неделя 3. Двоичная куча. Сортировка кучей

Устная часть

3.39. Пусть в двоичную кучу добавили n элементов. Сколько листьев у получившегося дерева?

3.40. Пусть в двоичную кучу в некотором порядке добавили все натуральные числа от 1 до 1 000 (каждое число было добавлено ровно один раз). Какое минимальное число может находиться на самом нижнем слое кучи?

3.41. Пусть в двоичную кучу добавили n элементов. Докажите, что в такой куче найдется не более, чем $\left\lceil \frac{n}{2^{h+1}} \right\rceil$ вершин с высотой h (*высота* — это расстояние от вершины до самого глубокого листа в поддереве этой вершины).

3.42. Придумайте способ модифицировать двоичную кучу на минимум, добавив новую операцию `decrease_key(v, x)` — изменить значение, находящееся в вершине v, на число x (гарантируется, что x не больше, чем старое значение). Данная операция должна работать за $\mathcal{O}(\log n)$. Можно ли реализовать аналогичную операцию `increase_key(v, x)`?

3.43. Придумайте структуру данных, позволяющую выполнять следующие операции за $\mathcal{O}(\log n)$:

- ▷ Добавить элемент
- ▷ Найти и удалить минимальный элемент
- ▷ Найти и удалить максимальный элемент

3.44. Пусть a — некоторый массив длины n, а b — массив a, отсортированный по неубыванию. Назовем *медианой* массива a элемент x, равный $b_{\frac{n}{2}}$. Придумайте структуру данных, позволяющую выполнять следующие операции за $\mathcal{O}(\log n)$:

- ▷ Добавить элемент
- ▷ Найти и удалить медиану

3.45. Дан массив длины n. Найдите в нем k минимальных элементов за $\mathcal{O}(n+k\log n)$.

3.46. Дан массив длины n. Найдите в нем k минимальных элементов за $\mathcal{O}(n+k\log k)$.

3.47. Приведите пример массива длины n, в котором все числа попарно различны, для которого сортировка кучей работает за $\mathcal{O}(n)$.

3.48. Назовем d-ичной кучей дерево, организованное следующем образом: у каждой вершины (кроме нижнего слоя) имеется ровно d детей. Какие номера имеют дети и родитель вершины с номером v? За какое время работают основные операции с кучей на таком дереве? Приведите оценку, зависящую от n и d.

3.49. Имеется d-ичная куча размера n. С ней выполняются m операций `extract_min` и k операций `decrease_key`. Выберите такое d, чтобы суммарное время работы было оптимально.

3.50. Имеются k отсортированных по неубыванию массивов, суммарный размер всех массивов равен n. Придумайте алгоритм, позволяющий все массивы в один отсортированный массив размера n за $\mathcal{O}(n \log k)$.

3.51. Пусть имеется двоичная куча. Придумайте алгоритм, позволяющий добавить в кучу k элементов за $\mathcal{O}(k + \log^2 n)$, где n — количество элементов в куче после добавления.

3.52. На лекции был рассмотрен код, строящий двоичную кучу без использования дополнительной памяти за $\mathcal{O}(n \log n)$:

```
for (i = 0...n - 1):
    sift_up(i)
```

Теперь попробуем поменять операцию `sift_up` на операцию `sift_down`, получим следующий код:

```
for (i = 0...n - 1):
    sift_down(i, n)
```

Приведите пример массива, на котором данный алгоритм не строит корректную двоичную кучу.

3.53. Докажите, что следующий алгоритм корректно строит кучу на заданном массиве:

```
for (i = n - 1...0):
    sift_down(i, n)
```

3.54. Докажите, что $\sum\limits_{i=0}^{\infty} \frac{1}{2^i} = 2$, а также, что $\sum\limits_{i=0}^{\infty} \frac{i}{2^i} = 2$. Данное доказательство пригодится в следующем задании.

3.55. Докажите, что код, приведенный в задании 2.53, строит двоичную кучу за $\mathcal{O}(n)$. При доказательстве можно считать факты из задания 2.54 доказанными.

3.56. Докажите или опровергните, что сортировка кучей является стабильной.

3.57. Даны две кучи с размерами n и m. Придумайте алгоритм, позволяющий слить данные кучи в одну кучу размера $n + m$ за $\mathcal{O}(n + m)$.

Неделя 4. Сборная солянка

Устная часть

4.58. Дано рекуррентное соотношение $T(n) = 3T\left(\frac{n}{2}\right) + n^2$. Можно считать, что $T(1) = 1$, либо выбрать более удобную базу индукции. Определите Θ-асимптотику данного соотношения и докажите ее по индукции.

4.59. На прямой живут n друзей, i-й друг живет в точке x_i. Друзья хотят встретиться в одной точке. Помогите им найти такую точку, чтобы суммарное пройденное расстояние было как можно меньше. Время работы: $\mathcal{O}(n \log n)$.

4.60. На прямой живут n друзей, i-й друг живет в точке x_i. Друзья хотят встретиться в одной точке. Помогите им найти такую точку, чтобы сумма квадратов пройденных расстояний была как можно меньше. Время работы: $\mathcal{O}(n \log n)$.

4.61. В задании 2.21 мы научились объединять k отсортированных массивов в один отсортированный массив за $\mathcal{O}(nk)$. Придумайте способ сделать это за $\mathcal{O}(n \log k)$.

4.62. Оцените время работы сортировки слиянием, если на каждом уровне рекурсии мы разбиваем массив не на две части, а на k. Приведите оценку, зависящую от n и k. Считайте, что объединить k отсортированных массивов в один отсортированный массив можно за $\mathcal{O}(nk)$.

4.63. Оцените время работы сортировки слиянием, если на каждом уровне рекурсии мы разбиваем массив не на две части, а на k. Приведите оценку, зависящую от n и k. Считайте, что объединить k отсортированных массивов в один отсортированный массив можно за $\mathcal{O}(n \log k)$.

4.64. Дан отсортированный массив длины n. Определите, можно ли разделить элементы массива на две группы таким образом, чтобы в каждой группе любые два числа отличались хотя бы на k. Время работы: $\mathcal{O}(n)$.

4.65. Дан массив длины n, состоящий из целых чисел. После этого поступают q запросов вида: «вычислить сумму $a_l + a_{l+1} + \ldots + a_r$». Научитесь отвечать на такие запросы за $\mathcal{O}(1)$. Разрешается использовать $\mathcal{O}(n)$ дополнительной памяти и выполнить некоторую предварительную обработку данных (далее мы будем называть это *препроцессингом*) за $\mathcal{O}(n)$.

4.66. Дан массив длины n, изначально заполненный нулями. После этого поступают q запросов вида: «прибавить к элементам $a_l, a_{l+1}, \ldots, a_r$ число x». Необходимо обработать все запросы и вывести итоговый массив. Время работы: $\mathcal{O}(n + q)$.

4.67. Дан массив длины n. Требуется найти подотрезок массива с максимальной суммой. Время работы: $\mathcal{O}(n)$.

4.68. Докажите, что не существует структуры данных, которая умеет выполнять следующие две операции асимптотически быстрее, чем за $\mathcal{O}(\log n)$:

- ▷ Добавить элемент
- ▷ Найти и удалить медианный элемент

4.69. Дан отсортированный массив длины n. Найдите в нем такие позиции x и y, что $x - y \leq k$ и $x - y$ максимально. Время работы: $\mathcal{O}(n)$.

4.70. Модифицируйте двоичную кучу таким образом, чтобы можно было выполнять операцию: «найти k-й минимум» за $\mathcal{O}(k \log k)$. Разрешается использовать $\mathcal{O}(n)$ дополнительной памяти.

4.71. Даны n точек на окружности, пронумерованных по часовой стрелке. Требуется найти две самые удаленные точки за $\mathcal{O}(n)$.

Неделя 5. Бинарный и тернарный поиск

Устная часть

5.72. Дан массив неотрицательных чисел длины n. Необходимо отвечать на запросы: какое максимальное количество чисел из начала массива можно взять, чтобы их сумма не превосходила x? Время на один запрос — $\mathcal{O}(\log n)$. Можно ли решить задачу таким же образом, если числа в массиве могут быть отрицательными?

5.73. Дан массив различных неотрицательных целых чисел длины n, отсортированный по возрастанию. За $\mathcal{O}(\log n)$ найдите минимальное неотрицательное целое число, которого в этом массиве нет.

5.74. Дан массив длины n, полученный циклическим сдвигом отсортированного по возрастанию массива. Все элементы массива попарно различны. Требуется за $\mathcal{O}(\log n)$ найти в массиве заданное число x. Можно ли решить задачу таким же образом, если числа в массиве могут повторяться?

5.75. Дан массив длины n, полученный приписыванием одного отсортированного по возрастанию массива в конец другого отсортированного по возрастанию массива. Все элементы массива попарно различны. Требуется за $\mathcal{O}(\log n)$ найти в массиве заданное число x.

5.76. Пусть выполняется целочисленный бинарный поиск с начальными значениями L и R. Придумайте алгоритм, позволяющий за $\mathcal{O}(\log(R-L))$ определить, могла ли пара чисел (l, r) быть границами бинарного поиска в некоторый момент времени.

5.77. Решите предыдущую задачу за $\mathcal{O}(1)$. Можно считать, что время работы любой арифметической операции с числами равно $\mathcal{O}(1)$.

5.78. Дан отсортированный по возрастанию массив длины n, а также число x. Найдите такую позицию p, что $a[p] = x$ за $\mathcal{O}(\log p)$ (обратите внимание, что нужно решение за $\mathcal{O}(\log p)$, а не за $\mathcal{O}(\log n)$).

5.79. В некоторой компьютерной игре имеется n видов ресурсов. Для постройки одного здания требуется a_i единиц i-го вида ресурсов для всех i от 1 до n. У вас есть b_i единиц ресурса i, а также g единиц золота. Одну единицу золота можно обменять на c_i единиц ресурса i-го вида. Определите, какое максимальное количество зданий вы сможете построить.

5.80. В выборах участвуют n кандидатов. Вы выяснили, что за i-го кандидата планируют проголосовать a_i человек. Вы хотите, чтобы кандидат с номером 1 победил в выборах (набрал строго больше голосов, чем все остальные кандидаты). За s рублей можно изменить мнение любого одного голосующего. Какое минимальное количество денег придется потратить на избирательную кампанию?

5.81. Пусть $f(x) = \sum\limits_i |a_i \cdot x + b_i|$ для некоторых (неизвестных вам) a_i и b_i. Можно ли найти минимум функции $f(x)$ при помощи тернарного поиска?

5.82. Дан отсортированный по возрастанию массив длины n. Требуется выбрать k чисел из массива таким образом, чтобы минимальная разность соседних двух выбранных чисел была как можно больше. То есть, формально, если вы выбрали числа $i_1, i_2, \ldots, i_k$, то нужно максимизировать $\min\limits_{j=1}^{k-1} a[i_{j+1}] - a[i_j]$.

5.83. Домик черепашки расположен в точке 0 числовой прямой. Однажды она узнала, что скоро вырастут n одуванчиков. Одуванчик с номером i вырастет в точке x_i в момент времени t_i, причем все x_i и t_i положительные, а так же $x_{i+1} > x_i$ и $t_{i+1} > t_i$. Черепашка выходит из дома в момент 0, она движется со скоростью не больше v. Чтобы съесть одуванчик, ей нужно остановиться около него на время d. За какое время она сможет съесть все одуванчики и вернуться домой?

5.84. Придумайте модификацию двоичной кучи, в которой операция `sift_up` делает $\mathcal{O}(\log \log n)$ сравнений элементов (очевидно, что количество обменов при этом уменьшить не получится).

5.85. Дан массив длины n, состоящий из нулей и единиц, причем первый элемент массива равен нулю, а последний элемент — единице. Найдите такую позицию i, что $a[i] = 0$ и $a[i+1] = 1$ за $\mathcal{O}(\log n)$.

5.86. При обычной реализации тернарного поиска на каждой итерации значение функции вычисляется в двух точках: m_1 и m_2. Придумайте способ выбирать точки m_1 и m_2 таким образом, чтобы на каждой итерации (кроме, возможно, первой) значение функции приходилось вычислять лишь в одной точке. При этом алгоритм все еще должен работать за $\mathcal{O}(\log n)$. Оцените точное количество итераций такого алгоритма (нужно привести точное значение основания логарифма).

Неделя 6. Стек. Очередь. Амортизационный анализ

Устная часть

6.87. Добавьте в стек и очередь операцию `get_sum()`, которая будет возвращать сумму всех элементов, находящихся в структуре данных. Время работы операции $\mathcal{O}(1)$. Дополнительная память $\mathcal{O}(1)$.

6.88. Добавьте в стек операцию `get_min()`, которая будет возвращать минимальный элемент, хранимый в стеке. Время работы операции $\mathcal{O}(1)$. Дополнительная память $\mathcal{O}(n)$, где n — количество элементов в стеке.

6.89. Придумайте способ реализовать очередь, используя два стека. Подсказка: один стек должен использоваться для добавления элементов, а второй — для удаления элементов. Все операции с очередью должны иметь амортизированное время работы $\mathcal{O}(1)$. Докажите эту оценку при помощи метода бухгалтерского счета или метода потенциалов.

6.90. Добавьте в очередь операцию `get_min()`, которая будет возвращать минимальный элемент, хранимый в очереди. Время работы операции $\mathcal{O}(1)$. Дополнительная память $\mathcal{O}(n)$, где n — количество элементов в очереди.

6.91. При помощи стека вычислите значение арифметического выражения, записанного в постфиксной записи (запись, в которой оператор записывается после операндов, например, выражение `4 - ((1 + 2) * 3)` будет записано как `4 1 2 + 3 * -`).

6.92. При помощи стека вычислите значение арифметического выражения, записанного в инфиксной записи (обычные выражения). Для простоты можно считать, что в выражении каждая операция взята в скобки, например `(4 - ((1 + 2) * 3))`.

6.93. Научитесь по выражению в инфиксной записи строить выражение в постфиксной записи.

6.94. Модифицируем стек, рассмотренный на лекции, который увеличивал размер массива в два раза при нехватке памяти для сохранения добавленного элемента. Во время операции `pop` будем уменьшать размер стека в два раза, если размер массива больше, чем количество хранимых элементов, хотя бы в четыре раза. Докажите при помощи метода бухгалтерского счета или при помощи метода потенциалов, что в таком стеке операции `push` и `pop` имеют амортизированное время работы $\mathcal{O}(1)$.

6.95. Дан массив из целых чисел. Для каждого числа требуется найти ближайшее слева число, меньшее данного (то есть формально для каждого i нужно найти такое $j < i$, что $a_j < a_i$ и j максимально). Суммарное время работы $\mathcal{O}(n)$.

6.96. Проанализируйте время работы стека, если расширение при нехватке памяти во время операции `push` происходит в $k > 1$ раз. Какое значение k лучше выбрать на практике?

6.97. Реализуйте очередь на базе расширяющегося и сужающегося массива с амортизированным временем работы всех операций $\mathcal{O}(1)$.

6.98. Добавьте в стек операцию `clear`, удаляющую все элементы из стека. Амортизированное время работы операции должно быть $\mathcal{O}(1)$. Докажите данную оценку.

6.99. Деком называется структура данных, которая позволяет выполнять следующие операции:

- ▷ `push_back(x)` — добавить элемент x в конец
- ▷ `push_front(x)` — добавить элемент x в начало
- ▷ `pop_back()` — удалить элемент из конца
- ▷ `pop_front()` — удалить элемент из начала
- ▷ `front()` — вернуть первый элемент
- ▷ `back()` — вернуть последний элемент

Реализуйте дек при помощи трех стеков, чтобы амортизированное время работы операций `push_back`, `pop_back`, `push_front` и `pop_front` было равно $\mathcal{O}(1)$. Докажите данную оценку.

6.100. Добавим в стек операцию `multipop(k)`, последовательно удаляющую k элементов из стека (гарантируется, что в стеке есть хотя бы k элементов). Докажите или опровергните, что амортизированное время работы всех операций равно $\mathcal{O}(1)$.

6.101. Помимо операции `multipop(k)` добавим операцию `multipush(k, x)`, последовательно добавляющую в стек k элементов, равных x. Докажите или опровергните, что амортизированное время работы всех операций равно $\mathcal{O}(1)$.

6.102. Битовый счетчик хранит целое положительное число в виде массива двоичных цифр. Изначально все цифры равны 0. Операция `increment` увеличивает счетчик на 1. Реализуйте операцию `increment` и докажите, что ее амортизированное время работы равно $\mathcal{O}(1)$.

6.103. Как изменится время работы, если добавить операцию `add(k)`, прибавляющую к счетчику 2^k?

6.104. Как изменится время работы, если добавить операцию `decrement`, уменьшающую счетчик на 1?

6.105. Добавьте битовому счетчику операцию `clear`, присваивающую ему 0. Амортизированная стоимость операции должна быть $\mathcal{O}(1)$.

Неделя 7. Связные списки. Pointer Machine.

Устная часть

7.106. Дан набор из n элементов, у каждого элемента есть ссылка на какой-то другой элемент. Проверьте, правда ли все элементы образуют один большой кольцевой список (менять ссылки нельзя). Время работы $\mathcal{O}(n)$, можно использовать $\mathcal{O}(1)$ дополнительной памяти.

7.107. Задан односвязный список, возможно, с циклами (для каждого элемента известен `node.next`). Предложите алгоритм, который находит длину цикла p, достижимого из заданной вершины v, а также длину пути перед циклом s, используя не более, чем $\mathcal{O}(1)$ дополнительной памяти. Время работы: $\mathcal{O}(s+p)$. Сам список менять нельзя.

7.108. Предложите способ развернуть односвязный список за $\mathcal{O}(n)$, используя $\mathcal{O}(1)$ дополнительной памяти.

7.109. Научитесь сливать два отсортированных односвязных списка в один за время $\mathcal{O}(n)$, используя $\mathcal{O}(1)$ дополнительной памяти.

7.110. Отсортируйте связный список за время $\mathcal{O}(n \log n)$, используя $\mathcal{O}(\log n)$ дополнительной памяти.

7.111. Отсортируйте связный список за время $\mathcal{O}(n \log n)$, используя $\mathcal{O}(1)$ дополнительной памяти.

7.112. Придумайте способ хранения связного списка, который позволит разворачивать его за $\mathcal{O}(1)$, сохранив возможность выполнять прочие операции.

7.113. Есть таблица размера $n \times n$. С ней производятся m операций следующего вида: вырезать прямоугольный кусок таблицы, развернуть его на 180 градусов, и вставить обратно. Выведите состояние таблицы после выполнения всех операций. Время работы: $\mathcal{O}(nm)$.

7.114. Реализуйте двоичную кучу в Pointer Machine Model.

7.115. Как сделать в Pointer Machine Model структуру данных, позволяющую выполнять двоичный поиск в связном списке (список отсортирован и не меняется).

7.116. Счетчик — это целое число, изначально равное нулю, с которым можно производить две операции: инкремент и декремент. Реализуйте полностью персистентный счетчик с $\mathcal{O}(1)$ времени на операцию и $\mathcal{O}(1)$ дополнительной памяти на каждую версию счетчика.

7.117. Придумайте, как реализовать частично персистентную двоичную кучу с временем на одну операцию $\mathcal{O}(\log n)$ и дополнительной памятью на каждую версию $\mathcal{O}(\log n)$.

7.118. Придумайте, как реализовать полностью персистентную двоичную кучу с временем на одну операцию $\mathcal{O}(\log n)$ и дополнительной памятью на каждую версию $\mathcal{O}(\log n)$.

7.119. Покажите как сделать в Pointer Machine Model структуру, которая позволяет делать двоичный поиск в списке (список отсортирован и не меняется, требуется отвечать на запросы: найти максимальный элемент не больше x).

Неделя 8. Система непересекающихся множеств

Устная часть

8.120. Добавьте в СНМ операции `getMin(x)`, `getMax(x)` и `getSum(x)`, вычисляющие минимум, максимум и сумму элементов множества, в котором лежит элемент с номером x. Время работы: $\mathcal{O}(\alpha(m, n))$.

8.121. В изначально пустой граф по одному добавляются q ребер. После каждого добавления нужно вычислить размер максимальной по количеству вершин компоненты связности графа. Время работы: $\mathcal{O}(q \cdot \alpha(m, n))$.

8.122. Дан пустой граф на n вершинах. Требуется выполнять запросы двух типов:

(a) Добавить ребро в граф.

(b) Найти число ребер в компоненте связности, в которой находится вершина x.

Время работы на одну операцию: $\mathcal{O}(\alpha(m, n))$.

8.123. Дан пустой граф на n вершинах. Требуется выполнять запросы двух типов:

(a) Добавить ребро в граф.

(b) Найти количество компонент связности, являющихся деревьями.

Время работы на одну операцию: $\mathcal{O}(\alpha(m, n))$.

8.124. Дан пустой граф на n вершинах. Требуется выполнять запросы двух типов:

(a) Добавить ребро в граф.

(b) Найти количество компонент связности, являющихся циклами.

Время работы на одну операцию: $\mathcal{O}(\alpha(m, n))$.

8.125. Пусть у всех элементов СНМ есть вес. Добавьте в СНМ две операции:

(a) Увеличить веса всех элементов, находящихся в одном множестве с элементом x, на d.

(b) Найти вес элемента x.

Время работы на одну операцию: $\mathcal{O}(\alpha(m, n))$.

8.126. Дан массив a, состоящий из n нулей. Требуется выполнять операции:

(a) Присвоить единицу i-му элементу массива.

(b) Найти количество непрерывных отрезков, состоящих из единиц.

Время работы на одну операцию: $\mathcal{O}(\alpha(m, n))$.

8.127. Дан массив a, состоящий из n нулей. Требуется выполнять операции:

(a) Присвоить единицу i-му элементу массива.

(b) Найти ближайший ноль к i-му элементу массива.

Время работы на одну операцию: $\mathcal{O}(\alpha(m, n))$.

8.128. Докажите, что $\log_a^*(n) = \Theta(\log_b^*(n))$, если итерированный логарифм определен для оснований a и b.

8.129. Дан массив a из n положительных чисел. Найдите отрезок $[l, r]$ с максимальной величиной $\left(\sum\limits_{i=l}^{r} a_i\right) \cdot \min\limits_{i=l}^{r} a_i$. Время работы — $\mathcal{O}(n \log n)$. При желании разрешается решать эту задачу без использования СНМ.

8.130. Дан массив a из n положительных чисел. Найдите отрезок $[l, r]$ с максимальной величиной $\left(\sum\limits_{i=l}^{r} a_i\right) \cdot \min\limits_{i=l}^{r} a_i$. Время работы — $\mathcal{O}(n \cdot \alpha(m, n))$.

8.131. Дан граф, из которого последовательно q раз удаляется по одному ребру. После каждого удаленного ребра требуется вывести количество компонент связности в графе. Время работы: $\mathcal{O}(q \cdot \alpha(m, n))$.

8.132. Функция Аккермана $A(m, n)$ задается следующим образом:

$$A(m,n) = \begin{cases} n+1, & \text{если } m = 0; \\ A(m-1, 1), & \text{если } n = 0; \\ A(m-1, A(m, n-1)), & \text{иначе.} \end{cases}$$

Обратная функция Аккермана $\alpha(m, n)$ задается следующим образом:

$$\alpha(m,n) = \min_{i \geq 1} A\left(i, \left\lfloor \frac{m}{n} \right\rfloor\right) \geq \log_2(n)$$

Докажите, что если $m = \Omega(n \log n)$, то $\alpha(m, n) = \mathcal{O}(1)$.

8.133. Докажите, что если $m = \Omega(n \log^* n)$, то $\alpha(m, n) = \mathcal{O}(1)$.

8.134. Докажите, что $\alpha(m, n) = \mathcal{O}(\log^* n)$.

8.135. Докажите амортизированную оценку $\mathcal{O}(\log n)$ для эвристики сжатия путей (без ранговой эвристики).

8.136. **При корректном решении данное задание стоит 5 баллов вне зависимости от номера выхода.** Докажите амортизированную оценку $\mathcal{O}(\log^{**} n)$ для СНМ с обеими эвристиками. Функция $\log^{**} n$ определяется по аналогии с $\log^* n$:

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

8.137. **При корректном решении данное задание стоит 10 баллов вне зависимости от номера выхода.** Докажите амортизированную оценку $\alpha(m, n)$ для СНМ с обеими эвристиками.

Неделя 9. Быстрая сортировка. K-я порядковая статистика

Устная часть

9.1.1 Алгоритмы

9.138. Приведите пример массива, на котором быстрая сортировка работает за $\Omega(n^2)$, если в качестве опорного элемента выбирается самый левый элемент отрезка.

9.139. Приведите пример массива, на котором быстрая сортировка работает за $\Omega(n^2)$, если в качестве опорного элемента выбирается самый правый элемент отрезка

9.140. Приведите пример массива, на котором быстрая сортировка работает за $\Omega(n^2)$, если в качестве опорного элемента выбирается центральный элемент отрезка (то есть `a[(l + r) / 2]`)

9.141. Решите ту же задачу при условии, что в качестве опорного элемента выбирается медиана из этих трех элементов.

9.142. Решите ту же задачу при условии, что в качестве опорного элемента выбирается среднее арифметическое этих трех элементов.

9.143. Как реализовать алгоритм быстрой сортировки таким образом, чтобы математическое ожидание времени работы алгоритма было равно $\mathcal{O}(n \log n)$ при условии, что в массиве могут быть повторяющиеся элементы?

9.144. Дан массив a длины n, а также массив p длины m. Для каждого i найдите p_i-ю порядковую статистику в массиве a. Суммарное время работы: $\mathcal{O}(m \log n + n)$.

9.145. Предположим, что злоумышленник знает, по какому принципу выбирается опорный элемент в быстрой сортировке (например, он знает `seed` и алгоритм, генерирующий случайные числа). Как он может составить массив, на котором алгоритм будет работать за $\Omega(n^2)$?

9.146. Как с помощью генератора случайных чисел получить случайную перестановку длины n? Нужно, чтобы каждая перестановка могла быть сгенерирована равновероятно (то есть с вероятностью $\frac{1}{n!}$). Напомним, что *перестановкой* длины n называется массив, в котором каждое из чисел от 1 до n встречается ровно один раз.

9.147. Что будет, если в алгоритме Блюма-Флойда-Пратта-Ривеста-Тарьяна заменить константу 5 на 3 или 7? Оцените асимптотику такого алгоритма.

9.1.2 Математика

9.148. Докажите, что для любых двух случайных величин y_1 и y_2, и некоторого вещественного числа x, выполнены следующие свойства:

- $E(y_1 + y_2) = E(y_1) + E(y_2)$,
- $E(x \cdot y_1) = x \cdot E(y_1)$.

9.149. *Перестановкой* называется набор из n различных целых чисел от 1 до n. *Неподвижной точкой* перестановки p называется такой индекс i, что $p_i = i$. Вычислите математическое ожидание количества неподвижных точек в случайной перестановке.

9.150. Вычислите математическое ожидание количества инверсий в случайной перестановке.

9.151. Докажите, что $\sum\limits_{i=0}^{\infty} \frac{1}{3} \cdot \left(\frac{2}{3}\right)^i \cdot i = \mathcal{O}(1)$.

Неделя 10. Динамическое программирование — 1

Устная часть

10.152. Котик прыгает с кочки 1 на кочку n, длина каждого прыжка может быть от 1 до k. У каждой кочки есть стоимость. Найдите путь с минимальной стоимостью за $\mathcal{O}(nk)$.

10.153. Решите предыдущую задачу за $\mathcal{O}(n)$.

10.154. Котик прыгает с кочки 1 на кочку n, длина каждого прыжка может быть 1 или 2. У каждой кочки есть стоимость. Найдите путь с минимальной стоимостью и количество путей с минимальной стоимостью за $\mathcal{O}(n)$.

10.155. Котик прыгает с кочки 1 на кочку n, длина каждого прыжка может быть от 1 до k. У каждой кочки есть стоимость. Найдите путь с минимальной стоимостью и количество путей с минимальной стоимостью за $\mathcal{O}(n)$.

10.156. Котик прыгает с кочки 1 на кочку n, длина каждого прыжка может быть 1 или 2. На каждой кочке написана буква. Найдите такой путь, чтобы строка, которую прочитает котик, посещая кочки, была лексикографически минимальной за $\mathcal{O}(n^2)$.

10.157. Черепаха ползет из клетки $(1, 1)$ в клетку (n, m), каждый раз перемещаясь в соседнюю справа или снизу клетку. В каждой клетке растет некоторое количество цветов. Посещая клетку, черепаха съедает все цветы, растущие в ней. Найдите максимальное **нечетное** количество цветов, которое она сможет съесть (пропускать цветы на пути нельзя). Время работы $\mathcal{O}(nm)$.

10.158. Черепаха ползет из клетки $(1, 1)$ в клетку (n, m), каждый раз перемещаясь в соседнюю справа или снизу клетку. В каждой клетке растет некоторое количество цветов. Посещая клетку, черепаха съедает все цветы, растущие в ней. Найдите количество различных путей черепахи, на которых она съест хотя бы k цветов. Время работы $\mathcal{O}(nmk)$.

10.159. Черепаха ползет из клетки $(1, 1)$ в клетку (n, m), каждый раз перемещаясь в соседнюю справа или снизу клетку. В каждой клетке записано целое число a_{ij}. Если это число положительное, то черепаха получает a_{ij} монет, а иначе — платит a_{ij} монет. Сколько монет нужно иметь черепахе в начале, чтобы добраться до конца пути таким образом, чтобы количество монет у нее всегда было неотрицательным. Время работы: $\mathcal{O}(nm \log C)$, где C — ответ.

10.160. У Васи есть калькулятор, который умеет выполнять три операции: прибавить 1, умножить на 2 и умножить на 3. Какое наименьшее количество операций нужно сделать, чтобы получить из числа 1 число n? Время работы $\mathcal{O}(n)$.

10.161. В очереди за билетами на концерт Моргенштерна стоят n школьников. Чтобы увеличить скорость движения очереди, два подряд идущих школьника могут договориться, и тот, что идет раньше в очереди купит два билета: на себя и на следующего. Школьник i тратит a_i секунд на покупку одного билета и b_i секунд на покупку двух билетов. За какое минимальное время все смогут купить билеты?

10.162. Вася любит ходить в кино и решать домашку по АиСД. Еще Вася любит разнообразие, поэтому он никогда не делает одно и тоже два дня подряд. Для n последовательных дней известно, есть ли в этот день интересное кино и есть ли в этот день интересная домашка. Какое минимальное число дней Вася будет сидеть без дела?

10.163. Посчитайте количество битовых последовательностей длины n, в которых не встречается подстрока 111. Время работы $\mathcal{O}(n)$.

10.164. Решите предыдущую задачу за $\mathcal{O}(\log n)$.

10.165. Пусть есть некоторая задача, которая решается при помощи динамического программирования. Известно, что $dp[1] = a_1, dp[2] = a_2, \ldots, dp[k] = a_k$. Также известно, что $dp[i] = \sum_{j=1}^{k} dp[i-j] \cdot b_j$ для всех $i > k$. Научитесь вычислять $dp[n]$ за $\mathcal{O}(k^3 \log n)$.

10.166. Дана строка s длины n и строка t длины m. За одну операцию можно удалить любой символ из строки s, добавить любой символ в любое место стоки s, либо поменять любой символ строки s. Найдите минимальное количество операций, за которое можно прекратить строку s в строку t. Время работы $\mathcal{O}(nm)$.

10.167. Как изменится решение предыдущей задачи, если у каждой операции есть своя стоимость и нужно вычислить, за какую минимальную стоимость можно преобразовать s в t?

10.168. Назовем оптимальное количество операций из предыдущих задач *редакционным расстоянием* между строками s и t. Даны две строки a и b. Найдите такую строку c, чтобы максимальное из редакционных расстояний: от c до a, и от c до b, было минимально возможным. Время работы $\mathcal{O}(nm)$.

10.169. Дан шаблон, состоящий из символов «`?`» и «`*`». Символ «`?`» можно заменить на произвольный символ, а символ «`*`» — на произвольную строку (возможно, пустую). Дана строка s, являющаяся шаблоном, а также строка t. Проверьте, можно ли строку s преобразовать в строку t при помощи некоторых замен. Время работы $\mathcal{O}(nm)$.

Неделя 11. Динамическое программирование — 2

Устная часть

11.170. У Пети есть n книг, отсортированных по названию. Книга i имеет высоту h_i. Петя хочет сложить книги в шкаф в отсортированном порядке: первые несколько книг положить на первую полку, следующие несколько книг — на вторую полку, и так далее. На каждую полку помещается не более, чем m книг. Высота полки равна максимальной высоте книги на полке. Помогите Пете распределить книги таким образом, чтобы суммарная высота полок была как можно меньше. Время работы $\mathcal{O}(nm)$.

11.171. Решите предыдущую задачу за $\mathcal{O}(n)$.

11.172. Дана строка s длины n. Вычислите для каждой пары (i, j) длину наибольшего общего префикса строк $s[i \dots n]$ и $s[j \dots n]$. Время работы $\mathcal{O}(n^2)$.

11.173. Дан массив длины n. Найдите наидлиннейшую подпоследовательность массива, в которой каждый следующий элемент делится на предыдущий. Время работы $\mathcal{O}(n^2)$.

11.174. Дан массив a длины n и массив b длины n. Найдите длину их наибольшей общей возрастающей подпоследовательности. Время работы $\mathcal{O}(n^3)$.

11.175. Решите предыдущую задачу за $\mathcal{O}(n^2)$.

11.176. Дан массив a длины n. Для каждого элемента a_i определите одно из трех:

(a) Элемент a_i не принадлежит ни одной НВП

(b) Элемент a_i принадлежит всем НВП

(c) Элемент a_i принадлежит хотя бы одной НВП

Время работы $\mathcal{O}(n \log n)$.

11.177. Дан массив a длины n. Все элементы попарно различны. Докажите, что либо длина наибольшей возрастающей подпоследовательности, либо длина наибольшей убывающей подпоследовательности не меньше, чем $\sqrt{n}$.

11.178. У колдуна есть n заклинаний и m единиц маны. Заклинание i тратит c_i единиц маны и наносит противнику d_i единиц урона. Убейте монстра, обладающего h единицами здоровья, потратив как можно меньше маны. Время работы: $\mathcal{O}(nm)$.

11.179. Петя и Вася выиграли на олимпиаде n призов, для каждого из которых известна его стоимость. Суммарная стоимость призов равна r. Друзья хотят разделить призы таким образом, чтобы разница в суммарной стоимости призов Пети и призов Васи была как можно меньше. Время работы: $\mathcal{O}(nr)$.

11.180. Решите предыдущую задачу за время $\mathcal{O}(2^{\frac{n}{2}} \cdot n)$.

11.181. Петя и Вася выиграли на олимпиаде $n = 2k$ призов, для каждого из которых известна его стоимость. Суммарная стоимость призов равна r. Друзья хотят разделить призы таким образом, чтобы каждому из них досталось ровно k

призов и чтобы разница в суммарной стоимости призов Пети и призов Васи была как можно меньше. Время работы: $\mathcal{O}(n^2 r)$.

11.182. Решите предыдущую задачу за время $\mathcal{O}(2^{\frac{n}{2}} \cdot n)$.

11.183. Имеется следующий код:

```
for (int s = mask; s > 0; s = (s - 1) & mask)
    print(s)
print(0)
```

Докажите, что данный код выводит все подмаски маски $mask$, упорядоченные по убыванию. Здесь «&» обозначает побитовое «И».

11.184. Имеется следующий код:

```
for (int mask = 0; mask < (1 << n); ++mask)
    for (int s = mask; s > 0; s = (s - 1) & mask)
        // Some work...
```

Докажите, что данные два цикла совершают $\mathcal{O}(3^n)$ итераций суммарно.

Неделя 12. Разные задачи на динамическое программирование

Устная часть

12.185. Задан массив длины n. Найдите в этом массиве **подпоследовательность** максимальной длины, которая сначала возрастает, а потом убывает. Время работы: $\mathcal{O}(n^2)$.

12.186. Задан массив длины n. Найдите в этом массиве **подпоследовательность** максимальной длины, которая является *пилообразной*. Последовательность называется *пилообразной*, если любой ее элемент либо строго меньше соседних элементов, либо строго больше соседних элементов. Время работы: $\mathcal{O}(n^2)$.

12.187. Имеется n предметов с весами w_i и стоимостями c_i. Нужно выбрать некоторое множество предметов, суммарный вес которых не превосходит W, максимизируя суммарную стоимость выбранных предметов. При этом запрещается брать в множество предметы, номера которых отличаются на 1: например, если в множество взят предмет с номером i, то запрещается брать предметы с номерами $i-1$ и $i+1$. Время работы: $\mathcal{O}(n \cdot W)$.

12.188. Огромному человекоподобному роботу опять нужно дойти из клетки $(0,0)$ в клетку (n,m). На этот раз он не очень торопится, поэтому ему не обязательно идти кратчайшим путем, но при этом он хочет сделать не более k шагов. Найдите число способов сделать это. Время работы: $O(nmk)$.

12.189. Дана последовательность чисел. Найдите максимальную по длине подпоследовательность, являющуюся арифметической прогрессией. Время работы: $O(n^3)$. Арифметической прогрессией называется последовательность вида: $a, a+d, a+2d, a+3d, \ldots$ для некоторых чисел a и d.

12.190. Решите предыдущую задачу за $O(n^2)$.

12.191. Даны два массива $p_1, p_2, \ldots, p_n$ и $q_1, q_2, \ldots, q_m$. Массивы p и q являются перестановками (все числа попарно различны и лежат в диапазоне от 1 до длины массива). Найти НОП массивов p и q за $\mathcal{O}(N \log N)$, где $N = \max(n, m)$.

12.192. Вычислить количество чисел, не превосходящих 10^n, в которых каждые две соседние цифры имеют НОД, равный 1.

12.193. Задано рекуррентное соотношение: $f_0 = 1$, $f_1 = 2$, $f_i = f_{i-1} + f_{i-2} + i$. Вычислите f_n за $\mathcal{O}(\log n)$.

12.194. Задано рекуррентное соотношение: $f_0 = 1$, $f_1 = 2$, $f_i = f_{i-1} + f_{i-2} + i^2$. Вычислите f_n за $\mathcal{O}(\log n)$.

12.195. Дана последовательность из n круглых, квадратных и фигурных скобок. Удалите минимальное число скобок, чтобы получилась правильная скобочная последовательность. Время $O(n^3)$.

12.196. Дана строка, состоящая из n символов. За одну операцию разрешается удалить любой символ строки. За какое минимальное количество операций можно получить из строки палиндром? Время работы: $\mathcal{O}(n^3)$.