1 Неделя 1

1.1 Устная часть

1. *Бордером* строки называется такое k, что $s[1..k] = s[|s| - k + 1..|s|]$. Число k является *периодом* строки, если для любого i, $s[i] = s[i + k]$. Докажите, что $b + p = |s|$, где p — минимальный период, а b — максимальный бордер.
2. Определим строку Туе-Морса: $T_n = t_0t_1t_2...t_{2^n-1}$, где $t_i = 0$, если двоичная запись числа i содержит четное число единиц, и $t_i = 1$ в противном случае. Доказать, что не существует двух равных как строки подстрок строки T_n, имеющих пересекающиеся вхождения в T_n.
3. Дана строка s. Посчитать матрицу $A : ||a_{ij}|| = \text{LCP}(s[i..n-1], s[j..n-1])$; $i, j \geq 0$ за $\mathcal{O}(|s|^2)$. (LCP — наибольший общий префикс двух строк). Предложите алгоритм, который вычисляет число различных подстрок на каждом префиксе строки s за $\mathcal{O}(|s|^2)$
4. Предложите алгоритм вычисления матрицы c_{ij} = число различных подстрок в $s[i..j]$ за $\mathcal{O}(|s|^2)$ с помощью матрицы A из предыдущего задания.
5. Вам заданы два массива чисел a и b. Найдите все подмассивы $a[i..i + |b| - 1]$ такие, что существует x, что $a[i] = b[1] + x, a[i + 1] = b[2] + x, \dots a[i + |b| - 1] = b[|b|] + x$ за время $\mathcal{O}(|a| + |b|)$.
6. Задана строка. Пусть $p_1[i]$ — максимальная длина палиндрома нечетной длины с центром в позиции i. $p_0[i]$ — аналогично для четной длины. Модифицировать алгоритм поиска z-функции для построения p_0 и p_1 за линейное время.
7. Как найти строку длины m в тексте длины n с использованием префикс-функции и $\mathcal{O}(m)$ дополнительной памяти?
8. Как найти строку длины m в тексте длины n с использованием z-функции и $\mathcal{O}(m)$ дополнительной памяти?
9. Назовем строку *простой*, если она лексикографически меньше всех своих суффиксов меньшей длины. Предложите алгоритм, который за $\mathcal{O}(|s|)$ проверяет, что строка s простая.
10. Предложите алгоритм, который по строке вычисляет массив q. $q[i]$ — такое максимальное число, что $s[1..q[i]] = s[i-q[i]+1..i] = s[i..i+q[i]-1]$. Время работы алгоритма $\mathcal{O}(|s| \log |s|)$.
11. Вам заданы строки s и t длины n. А также перестановка π, тоже длины n. Найдите, какое минимальное число раз нужно применить перестановку π к строке s, чтобы получить строку t. Время работы $\mathcal{O}(n)$. (Считайте, что арифметические операции с числами вида «число способов» выполняются за $\mathcal{O}(1)$.)
12. Докажите, что число различных подстрок-палиндромов в строке длины n не более n.

2 Неделя 2

2.1 Устная часть

13. Задан набор строк $s_1, s_2, \dots, s_n$. Докажите три утверждения:

(a) Если взять любую строку t, то $|\{\text{LCP}(s_i, t) | 1 \le i \le n\}| = \mathcal{O}\left(\sqrt{\sum |s_i|}\right)$, где $\text{LCP}(s_i, t)$ — длина наибольшего общего префикса s_i и t.

(b) У любой вершины в боре, в который добавили все слова s_i, $\mathcal{O}\left(\sqrt{\sum |s_i|}\right)$ вершин-предков, которые являются терминальными.

(c) У любой вершины в боре, в который добавили все слова s_i, $\mathcal{O}\left(\sqrt{\sum |s_i|}\right)$ вершин-предков, у которых хотя бы два ребенка.

14. Дано n строк. Найдите количество пар индексов $(i, j), 1 \le i, j \le n$, что $s_i s_j$ является палиндромом, за время $\mathcal{O}(\sum |s_i|)$.

15. Докажите, что если строки s и t таковы, что $st = ts$, то найдется такая строка p, что $s = p^i$ и $t = p^j$ для некоторых i и j.

16. Строки Фибоначчи. Определим $F_0 = \varepsilon$, $F_1 = b$, $F_2 = a$, $F_n = F_{n-1}F_{n-2}$. Докажите, что существует k такое, что для $n \ge k$ выполнено: F_n^2 — префикс F_{n+2}.

17. Строки Фибоначчи. Докажите, что существует k такое, что если $n \ge k$, то строка $F_n[1...|F_n| - 2]$ — палиндром.

18. У вас есть строка из цифр длины n. Вы можете порезать строчку на k непустых подстрок. Максимизируйте сумму чисел, полученных из этих подстрок, за линейное время. Предложите алгоритм, как получить это число в той же системе счисления, в которой эти числа записаны в строке.

19. Возьмем все простые строки над алфавитом $\{a, b\}$ длин, являющихся делителями n, упорядочим их лексикографически и сконкатенируем. Будем рассматривать получившуюся конкатенацию как зацикленную строку. Докажите, что в такой циклической строке все подстроки длины n различны, и множество всех этих подстрок совпадает с множеством строк длины n.

20. Найдите наибольшую общую подстроку двух строк с помощью хеширования за время $\mathcal{O}(n \log n)$, где n — размер входных данных.

21. Вам задан текст t и слово w. Для каждой позиции в строке t определите, существует ли перестановка букв алфавита, что если ее применить к слову, то в этой позиции будет вхождение этого слова за время $\mathcal{O}(|t| + |w|)$ с помощью хеширования.

22. Предыдущая задача, но с нулевой вероятностью ошибки.

23. Посчитать число различных подстрок палиндромов в строке длины n за линейное время $\mathcal{O}(n)$.

24. Предыдущая задача, но с нулевой вероятностью ошибки.

25. Задано две матрицы $A_{n \times n}$ и $B_{m \times m}$ $(m \le n)$, состоящие из букв. Найдите все вхождения матрицы B в матрицу A за время $\mathcal{O}(n^2 + m^2)$.

26. Задано две матрицы $A_{n \times n}$ и $B_{m \times m}$ $(m \le n)$, состоящие из букв. Найдите все вхождения матрицы B в матрицу A за время $\mathcal{O}(n^2 + m^2)$ **без использования хеширования**.

3 Неделя 3

3.1 Устная часть

27. Задано множество слов w_i. Построив автомат Ахо-Корасик для множества строк, научитесь за $\mathcal{O}(|t| + \sum |w_i|)$ отвечать на вопрос: сколько раз каждое слово входит в текст как подстрока, в какой позиции было первое вхождение каждого из слов, в какой последнее?
28. Задано множество слов w_i. Построив автомат Ахо-Корасик для множества строк за время $\mathcal{O}(\sum |w_i|)$, научитесь за $\mathcal{O}(|t| \log |t|)$ отвечать на вопрос: сколько слов входит в текст как подстрока, когда было самое первое вхождение какого-нибудь из слов?
29. Посчитать количество строк длины n на алфавите размера m, не содержащих никакую из заданных строк $s_1, s_2, \ldots, s_k$ как подстроку, за полиномиальное от длины входных данных время.
30. Задан набор строк $s_1, s_2, \ldots, s_k$. Предложите алгоритм, который вычисляет число строк длины n над алфавитом размера m, которые покрыты словами из s, за полиномиальное от размера входных данных время. Строка покрыта множеством слов, если для любой позиции строки существует вхождение какого-нибудь слова из множества, что это вхождение содержит эту позицию строки.
31. Задан набор строк $s_1, s_2, \ldots, s_k$. Предложите алгоритм, который вычисляет k-ю в лексикографическом порядке строку среди строк длины n над алфавитом размера m таких, что они содержат хотя бы одно из слов из s, за полиномиальное от n, m и размера входных данных время.
32. Предложите модификацию алгоритма Ахо-Корасик, которая позволяет по тексту t за время $\mathcal{O}(|t|)$ найти все состояния, соответствующие каждому префиксу t, но при этом хранит только суффиксные ссылки и переходы в боре.
33. Дано два бора A и B. Для всех вершин u в A найти самую глубокую вершину v в B, соответствующую суффиксу u (префикс-функция бора в боре). Время работы $\mathcal{O}(|A|+|B|)$.
34. Задано n шаблонов. Существует ли бесконечная вправо строка, которая не содержит ни одного шаблона как подстроки? Определите это за полиномиальное от размера входа время.
35. Задано n шаблонов. Существует ли бесконечная в обе стороны строка, которая не содержит ни одного шаблона как подстроки? Определите это за полиномиальное от размера входа время.
36. Задан набор строк $s_1, s_2, \ldots, s_n$ суммарной длины S. Предложите алгоритм, который один раз ответит на много запросов вида: $\text{get}(l, r, t) = \sum_{i=l}^{r} f(t, s_i)$, где $f(t, w)$ — число вхождений слова w в текст t. Время работы: $\mathcal{O}(S + (n + \sum |t_i|) \log S)$.

4 Неделя 4

4.1 Устная часть

37. Предложите алгоритм, который, имея суффиксный массив и LCP, считает число различных подстрок в строке за $\mathcal{O}(n)$.
38. Предложите алгоритм, который, имея суффиксный массив и LCP, считает число различных подстрок на каждом суффиксе строки за $\mathcal{O}(n \log n)$. Бонус: как это сделать за $\mathcal{O}(n)$?
39. Предложите алгоритм, который, имея суффиксный массив и LCP, считает число различных подстрок на каждом префиксе строки за $\mathcal{O}(n \log n)$. Бонус: как это сделать за $\mathcal{O}(n)$?
40. Рассмотрим алгоритм поиска LCP с лекции (Kasai, Arimura, Arikawa, Lee, Park) для суффиксного массива циклических сдвигов строки без конечного символа. Покажите контрпример, на котором он работает некорректно. Покажите, как найти LCP для соседних в лексикографическом порядке циклических сдвигов строки.
41. Задана строка и перестановка. Предложите алгоритм без использования хеширования, который за $\mathcal{O}(n)$ проверяет, что перестановка является суффиксным массивом строки.
42. Задана строка. Для каждого суффикса $s[i..n]$ определите, больше он лексикографически суффикса $s[i+1..n]$ или нет, за время $\mathcal{O}(n)$.
43. Задан суффиксный массив, построенный по строке, состоящей из символов двоичного алфавита. Восстановите любую двоичную строку с таким суффиксным массивом за $\mathcal{O}(n)$.
44. Задана строка длины n. Найдите самую длинную подстроку, которая имеет хотя бы два непересекающихся вхождения в строку. Время: построение суффиксного массива + $\mathcal{O}(n)$.
45. Задана строка длины n и число k. Найдите самую длинную подстроку, которая имеет хотя бы k вхождений в строку. Время: построение суффиксного массива + $\mathcal{O}(n)$.
46. Задана строка длины n и число k. Посчитать число различных подстрок строки, которые имеют хотя бы k вхождений в строку, за $\mathcal{O}(n \log n)$.
47. Заданы строки суммарной длины n. Предложите алгоритм, который с помощью хеширования за $\mathcal{O}(n \log^2 n)$ находит наидлиннейшую подстроку, входящую во все строки.
48. Заданы строки суммарной длины n. Предложите алгоритм, который с помощью построения суффиксных массивов суммарной длины $\mathcal{O}(n)$ и $\mathcal{O}(n)$ дополнительного времени находит наидлиннейшую подстроку, входящую во все строки.
49. Задана строка s длины n. Предложите алгоритм, отвечающий на запрос в онлайн: «Заданы i и j: Сколько раз $s[i..j]$ входит в s как подстрока?». Время: построение суффиксного массива + $\mathcal{O}(n)$ предпросчёта, $\mathcal{O}(\log n)$ на запрос.

5 Неделя 5

5.1 Устная часть

50. Предложите линейный алгоритм построения суффиксного дерева по заданному суффиксному массиву и LCP.

51. Решите с помощью суффиксного дерева за линейное время: Задана строка длины n. Найдите самую длинную подстроку, которая имеет хотя бы два непересекающихся вхождения в строку.
52. Решите с помощью суффиксного дерева за линейное время: Задана строка длины n и число k. Найдите самую длинную подстроку, которая имеет хотя бы k вхождений в строку.
53. Решите с помощью суффиксного дерева за линейное время: Задана строка длины n и число k. Вычислите число различных подстрок, которые имеют хотя бы k вхождений в строку.
54. Заданы строки $s_1, s_2, \ldots, s_n$. Для каждой строки найдите её минимальный префикс, который не является префиксом никакой другой строки, или сообщите, что таких префиксов не существует. Решите за линейное от размера входных данных время.
55. Заданы строки $s_1, s_2, \ldots, s_n$. Для каждой строки найдите её минимальную подстроку, которая не является подстрокой никакой другой строки, или сообщите, что таких подстрок нет. Решите за линейное от размера входных данных время.
56. Задана строка s. Ответьте на запросы в online: задано k подстрок строки s, найдите две из них, у которых максимальный общий префикс. Время работы: $\mathcal{O}(k \log (n + k))$ на запрос и $\mathcal{O}(n)$ предпросчёта.
57. Задана строка s. Пусть $k(w)$ — число вхождений w в s. Найдите число пар строк (x, y): x — подстрока y, y — подстрока s, $k(x) = k(y)$. Решите за $\mathcal{O}(|s|)$ с помощью суффиксного дерева.

6 Неделя 6

6.1 Устная часть

58. Заданы целые числа L, R и строка s. Предложите алгоритм, вычисляющий число различных подстрок длины от L до R в каждом префиксе строки s с использованием суффиксного дерева за $\mathcal{O}(|s|)$.
59. Решите с помощью суффиксного дерева за линейное время: Заданы строки суммарной длины n. Предложите алгоритм, который находит наидлиннейшую подстроку, входящую во все строки.
60. Задана строка s длины n. Ответьте на запросы $\langle L, R\rangle$ в online за время $\mathcal{O}(\log n)$ с предпросчетом за время $\mathcal{O}(n)$: найти максимальную подстроку-палиндром в $s[L..R]$.
61. Решите с помощью суффиксного дерева за линейное время: Пусть $Q(s)$ — множество всех подстрок строки s. Заданы строки s и t. Посчитайте размер множества $\{xy | x \in Q(s) \wedge y \in Q(t)\}$.
62. Решите с помощью суффиксного дерева. Задана строка s длины n. Предложите алгоритм, отвечающий на запрос в онлайн: «Заданы i и j: Сколько раз $s[i..j]$ входит в s как подстрока?». Время работы: $\mathcal{O}(\log n)$ на запрос и $\mathcal{O}(n \log n)$ предпросчёта.

7 Неделя 7

7.1 Устная часть

Предъявите decision-версию (с выходными данными «Да»/«Нет») и предложите полиномиальное сведение к ней:

63. CLIQUE (найти максимальную клику);
64. MAX-SAT (найти набор значений переменных, удовлетворяющий наибольшее число клозов в КНФ-формуле);
65. HAMPATH (найти гамильтонов путь в графе);
66. 3-COLORING (найти раскраску графа в три цвета).

Постройте сведение:

67. HAMPATH $\leq_p$ HAMCYCLE (гамильтонов цикл в графе) и HAMCYCLE $\leq_p$ HAMPATH;
68. 3-COLORING $\leq_p$ 3-SAT;
69. 3-SAT $\leq_p$ NAE-3-SAT (Not All Equal 3-SAT: в каждом клозе хотя бы одна истина и хотя бы одна ложь) и NAE-3-SAT $\leq_p$ 3-SAT;
70. NAE-3-SAT $\leq_p$ MAX-CUT (разделить множество вершин взвешенного графа на две доли, максимизировать суммарный вес рёбер между долями) и MAX-CUT $\leq_p$ NAE-3-SAT;
71. HAMPATH $\leq_p$ SAT.

Докажите NP-полноту decision-версий:

72. VERTEX-COVER (найти минимальное вершинное покрытие графа);
73. MAX-3-SAT (найти набор значений переменных, удовлетворяющий наибольшее число клозов в 3-КНФ-формуле);
74. EXACT-COVER (дано множество U, а также n его подмножеств $U_i \in U$; выбрать k и индексы $i_1, i_2, \ldots, i_k$, чтобы U_{i_j} попарно не пересекались, а в объединении давали U);
75. K-SPAN-TREE (проверить, что в неориентированном графе есть остовное дерево, в котором степень каждой вершины не более k).

8 Неделя 11

8.1 Устная часть

76. Докажите, что рассмотренные на лекции 2-приближённые алгоритмы не являются $(2-\varepsilon)$-приближёнными ни для какого $\varepsilon > 0$:

    - для задачи о вершинном покрытии;
    - для задачи коммивояжёра с условием неравенства треугольника.

77. Предложите алгоритм для поиска набора, удовлетворяющего не менее $\frac{7}{8}$ всех клозов 3-SAT, *среднее* время работы которого *на любом тесте* полиномиально. В каждом клозе ровно три различные переменные.
78. Предложите алгоритм для поиска набора, удовлетворяющего не менее $\frac{7}{8}$ всех клозов 3-SAT, *худшее* время работы которого полиномиально. В каждом клозе ровно три различные переменные.

Докажите или опровергните, что следующие алгоритмы являются X-приближёнными для некоторой константы X:

80. Алгоритм построения минимального вершинного покрытия: каждый раз добавлять в множество вершину, покрывающую максимальное число ещё не покрытых рёбер, пока все рёбра не станут покрытыми.
81. Алгоритм построения максимальной клики: каждый раз удалять вершину минимальной степени, пока не получится полный граф.
82. Алгоритм решения задачи о рюкзаке с неограниченным числом предметов каждого типа: отсортировать типы по отношению стоимости к весу и добавлять в таком порядке, если можно добавить (если добавить предмет не удалось, алгоритм не останавливается, а продолжается для следующих типов).

Предложите X-приближённые полиномиальные алгоритмы для решения следующих задач:

83. $X = 2$: построить во взвешенном неориентированном графе путь минимального веса, проходящий по каждому ребру хотя бы один раз.
84. $X = 2$: разложить n предметов с весами $w_1, w_2, \ldots, w_n$ ($w_i \leq S$) в минимальное число коробок, чтобы вес каждой коробки был не более S.
85. $X = 2$: раскрасить планарный граф в минимальное число цветов.
86. $X = 2$: в ориентированном графе выбрать максимальное по мощности множество рёбер такое, что полученный подграф не содержит циклов.
87. $X = 2$: разделить множество вершин неориентированного графа на две доли, чтобы число рёбер между долями было максимально.
88. $X = \frac{3}{2}$: задача коммивояжёра с условием неравенства треугольника.
89. $X = K$: во взвешенном неориентированном графе выделено K вершин. Построить связный подграф минимального веса, содержащий все выделенные вершины.
90. $X = \mathcal{O}(\sqrt{n})$: дано семейство подмножеств множества U, $|U| = n$. Выбрать из него максимальное число попарно непересекающихся подмножеств.

9 Неделя 12

9.1 Устная часть

91. Задачу раскраски графа в k цветов можно рассматривать, как разбиение всех вершин графа на k независимых подмножеств. Предложите контрпример к следующему алгоритму нахождения раскраски: k раз находим максимальное независимое множество, красим его в очередной цвет, удаляем из графа.

92. Для задачи 3-SAT с n переменными найдите любое назначение переменных, которое удовлетворяет все дизъюнкты за время $\mathcal{O}^*(c^n)$, где $c \approx 1.8393$ — максимальный корень уравнения $x^3 = x^2 + x + 1$.
93. Задан неориентированный граф из n вершин. Требуется узнать, существует ли вершинное покрытие мощности не более k за время $\mathcal{O}(2^k \operatorname{poly}(n))$.
94. Задан неориентированный граф из n вершин. Требуется узнать, существует ли вершинное покрытие мощности не более k за время $\mathcal{O}(\phi^k \operatorname{poly}(n))$, где $\phi = \frac{\sqrt{5}+1}{2}$.
95. Задан неориентированный граф из n вершин и число k. Докажите, что верно хотя бы одно из двух:
    - Существует простой путь из k вершин.
    - Существует раскраска графа в k цветов.

10 Неделя 13

11 Неделя 14

11.1 Устная часть

Здесь и далее просто сложность алгоритма подразумевает, что операции с числами делаются за $\mathcal{O}(1)$, а битовая сложность учитывает время работы операций.

96. Пусть алгоритм Евклида сделал n шагов на паре чисел $u > v > 0$. Сформулируйте точные нижние оценки на u и v в зависимости от n. Например, если $n = 1$, то $u \geq 2$ и $v \geq 1$.
97. Покажите, что битовая сложность стандартного алгоритма Евклида составляет $\mathcal{O}(\log^2 n)$, где $n = \max(a, b)$.
98. Предложите модификацию алгоритма Евклида, которая использует только операции сложения, вычитания, сравнения и умножения/деления на степень двойки. Время работы — $\mathcal{O}(\log n)$ таких операций. (Поскольку все такие операции имеют битовую сложность $\mathcal{O}(\log n)$, то в итоге выйдет также $\mathcal{O}(\log^2 n)$.)
99. То же самое, только для **расширенного** алгоритма Евклида.
100. Для заданных положительных a, b и c найдите число решений диофантова уравнения $ax + by = c$, для которых $x, y \geq 0$, за $\mathcal{O}(\log \max(a, b, c))$ операций с числами.
101. Найдите число пар (x, y) таких, что $1 \leq x \leq n$, $y > 0$ и $\frac{y}{x} \leq \frac{a}{b}$, за $O(\log \max(n, a, b))$.
102. Заданы целые числа m, x, l, r $(0 \leq l \leq r < m)$. Найдите минимальное $k \geq 0$, что $(k \cdot x) \bmod m \in [l, r]$, за $\mathcal{O}(\log m)$.
103. Каков критерий существования решения и алгоритм восстановления числа в КТО, если убрать требование взаимной простоты модулей m_1 и m_2?
104. Покажите, как найти обратные по модулю m к числам $1, 2, \ldots, n$ за $\mathcal{O}(n)$ операций с числами.

105. Покажите, как в алгоритме Миллера-Рабина в случае, когда мы нашли нетривиальный корень из 1 по модулю n, выдать делитель n.
106. Тест Лукаса на простоту: пусть известна факторизация $n-1$. Докажите, что n простое $\iff$ существует взаимнопростое a, что $a^{n-1} = 1 \pmod n$, $a^{\frac{n-1}{q}} \neq 1 \pmod n$, где q — любой простой делитель $n-1$. Составьте на основе этого утверждения алгоритм проверки n на простоту.
107. Зная факторизацию $n = p_1^{a_1} \dots p_k^{a_k}$, найдите сумму всех чисел от 0 до $n-1$, взаимнопростых с n, за $\mathcal{O}(k)$ операций с числами.
108. Пусть $n = pq$, где p и q — различные простые числа. Покажите, как, зная n и $\varphi(n)$, найти p и q.
109. Докажите, что $\varphi(n) = \Omega(n^{\frac{1}{2}})$.
110. Докажите, что $\varphi(n) = \Omega(\frac{n}{\log n})$.
111. Докажите, что $\varphi(n) = \Omega(\frac{n}{\log \log n})$.

12 Неделя 15

12.1 Устная часть

Здесь и далее просто сложность алгоритма подразумевает, что операции с числами делаются за $\mathcal{O}(1)$, а битовая сложность учитывает время работы операций.

112. В схеме RSA опасно брать простые, близкие друг к другу. Покажите, как факторизовать $n = pq$ за $\mathcal{O}(|p-q| \cdot \mathrm{poly}(\log n))$.
113. Нельзя брать простые, даже менее близкие друг к другу. Факторизуйте $n = pq$ за $\mathcal{O}(\mathrm{poly}(\log n))$, если известно, что $|p-q| < n^{\frac{1}{4}}$.
114. Предположим, что Алиса отправила одно и то же сообщение m, используя один и тот же модуль $n = pq$, но разные публичные экспоненты e_A и e_B такие, что $\gcd(e_A, e_B) = 1$. Покажите, как Еве, зная $m^{e_A} \bmod n$ и $m^{e_B} \bmod n$, восстановить m.
115. В RSA часто используется публичная экспонента e небольшого размера и с небольшим числом единичных битов, например, 3 или $65537 = 2^{16} + 1$. Как это помогает ускорить шифрование? При $e = 3$ посылка одного сообщения трем разным адресатам (с разными модулями) приводит к возможности расшифровки. Как?
116. Решите задачу дискретного логарифма (нахождение x: $a^x = b \pmod n$) за $O(\sqrt{x})$.
117. Решите задачу дискретного логарифма для простого модуля p вида $2^k + 1$ за $\mathcal{O}(\mathrm{poly}(k))$.
118. Предполагая, что мы знаем факторизацию $p-1$, обобщите решение предыдущей задачи для любого простого модуля p за $\mathcal{O}(\sqrt{r}\,\mathrm{poly}(\log p))$, где r — максимальный простой делитель $p-1$.
119. Зафиксируем простое p и два остатка g и h. Рассмотрим хеш-функцию $h(x, y) = g^x h^y \bmod p$. Как, найдя коллизию в такой хеш-функции, решить задачу дискретного логарифма? Предложите альтернативное решение задачи дискретного логарифма за $\mathcal{O}(\sqrt{p})$ на основе этого факта.

13 Неделя 16

13.1 Устная часть

120. Дан набор чисел $\{a_i\}$, $0 \le a_i \le M$. Найдите количество арифметических прогрессий длины 3, которые можно составить из этих чисел, за $\mathcal{O}(M \log M)$.

121. Изучите, что произойдет, если размер массива 2^k для преобразования Фурье меньше, чем число коэффициентов в результате ($\deg A + \deg B - 1 > 2^k$). Как получившиеся коэффициенты можно выразить через коэффициенты настоящего произведения многочленов?

122. Сведите к одному умножению многочленов. Даны два вектора $a[n]$ и $b[m]$ ($n \le m$). Вычислите скалярные произведения a и всех подстрок b длины n.

123. Сведите к одному умножению многочленов. Даны две строки p и t из символов 0 и 1 ($|p| \le |t|$). Найдите расстояние Хэмминга между p и всеми подстроками t длины $|p|$.

124. Для заданных n различных чисел $x_0, x_1, \ldots, x_{n-1}$ постройте многочлен n-й степени, который имеет корни только в заданных n точках, за $\mathcal{O}(n \log^2 n)$.

125. Рассмотрим $C(x) = A(x) + iB(x)$, где A и B — многочлены одинаковой длины с чисто вещественными коэффициентами, а i — мнимая единица. Покажите, как восстановить $DFT(A)$ и $DFT(B)$ по одному вызову $DFT(C)$.

126. Модифицируйте алгоритм FFT для случая, когда размер массива является степенью тройки ($n = 3^k$). Какое будет рекуррентное соотношение для времени работы и само время работы в таком случае?

127. Пусть заданы значения многочлена $A(x)$ степени n для всех $x = 0, 1, \ldots, n$. Для заданного числа $t > n$ вычислите $A(t)$ за $\mathcal{O}(n)$.

128. Двумерная интерполяция Лагранжа: по заданным наборам различных чисел $\{x_i\}_{i=0}^{n-1}$ и $\{y_j\}_{j=0}^{m-1}$ постройте вспомогательные многочлены

$$\varphi_{i,j}(x_k, y_l) = \begin{cases} 1, & i = k \wedge j = l \\ 0, & i \ne k \vee j \ne l \end{cases}$$

в которых степень x меньше n, а степень y меньше m. Постройте многочлен $A(x, y)$, что $A(x_i, y_j) = z_{i,j}$, за $\mathcal{O}((nm)^2)$.

129. Найдите $\gcd(A(x), (x+a)^{\frac{p-1}{2}} - 1)$ за $\mathcal{O}(|A|^2 \log p)$.

130. Дан многочлен $A(x)$ и число z. Вычислите $A(z^0), A(z^1), \ldots, A(z^{n-1})$ за $\mathcal{O}(n \log n)$.

131. Задан многочлен $A(x)$. Пусть мы знаем многочлен $B(x)$, что $A(x) \cdot B(x) - 1$ делится на x^k. Найдите такой многочлен $C(x)$, что $A(x) \cdot C(x) - 1$ делится на x^{2k}, за $\mathcal{O}(k \log k)$.

132. Задан многочлен $A(x)$, $a_0 \ne 0$. Найдите такой многочлен $B(x)$, что $A(x) \cdot B(x) = 1 + x^n Q(x)$ для некоторого многочлена $Q(x)$, за $\mathcal{O}(n \log n)$.

133. Используя предыдущее задание, вычислите $A(x) \bmod B(x)$ за $\mathcal{O}(n \log n)$.

134. Двумерное преобразование Фурье: двумерный массив a_{ij} размера $n \times n$ ($n = 2^k$) задает коэффициенты многочлена от двух переменных $A(x, y)$. Вычислите массив $\widetilde{A}_{s,t} = A(\omega_n^s, \omega_n^t) = \sum\limits_{i=0}^{n-1} \sum\limits_{j=0}^{n-1} a_{ij} \omega_n^{si} \omega_n^{tj}$ за $\mathcal{O}(2^{2k}k)$.

135. Используя двумерное преобразование Фурье, найдите скалярное произведение между матрицей $A_{n \times n}$ и всеми подматрицами размера $n \times n$ матрицы $B_{m \times m}$ за $\mathcal{O}(m^2 \log m)$.