Разбор задач областной олимпиады 2019

...

Авторы:
Никита Лесников
Никита Сечко

Typ 1

Задача 1. Олимп-Сити

70 баллов ($L \le R \le 10^6$)

Для получения **70** баллов надо было построить последовательность, описанную в условии, и найти сумму на соответствующем отрезке **[L, R]**.

1, 1, 2, 1, 2, 3, 1, 2, 3, 4, 1, 2, 3, 4, 5, 1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6, 7, 1, 2, 3, 4, 5, 6, 7, 8, ...

Строить данную последовательность требовалось за **O(R)**.

Чтобы этого добиться, записываем в конец некоторого, изначально пустого, массива сначала все натуральные числа от **1** до **1**, затем все натуральные числа от **1** до **2**, потом от **1** до **3**, ... , от **1** до **n**, пока длина данного массива не превышает **R**.

90 баллов ($R - L \le 10^6$)

Рассмотрим конечную последовательность:

1, 1, 2, 1, 2, 3, 1, 2, 3, 4, 1, 2, 3, 4, 5, ... , 1, 2, 3, 4, 5, 6, 7, 8, ..., n.

Нетрудно видеть, что она состоит из ровно n подпоследовательностей натуральных чисел от **1** до **i**. Где **i** принимает значения от **1** до **n**. Каждая такая подпоследовательность содержит ровно **i** чисел. Тогда нетрудно видеть, что вся данная последовательность содержит ровно $S(n) = 1 + 2 + 3 + ... + n = n*(n + 1)/2$ чисел. Тогда перебором найдём наименьшее n, для которого $L \le S(n)$. Тогда нетрудно видеть, что на позиции **L** будет находится число $L - S(n - 1)$. Таким образом начиная с позиции **L** и до позиции **R** восстановим всю последовательность от **L** до **R**. Асимптотика этого решения $O(max(L^{1/2}, R - L))$.

100 баллов

Рассмотрим n и $S(n)$ из предыдущего пункта. Заметим, что в данной задаче не обязательно восстанавливать последовательность. Также, заметим, что некоторые подпоследовательности чисел от 1 до i полностью входят в отрезок $[L ; R]$, и только две такие подпоследовательности входят в отрезок частично, и это подпоследовательности, в которых содержатся L и R:

$$L - S(n - 1), L - S(n - 1) + 1, \dots n, \quad 1, \dots (n + 1), \quad 1, \dots (n + 2), \quad \dots , \quad 1, 2, 3, \dots, R - S(m - 1).$$

Где m, по аналогии с n, является наименьшим таким натуральным числом, для которого $R \le S(m)$. Тогда для всех i от n до $(m - 1)$ сумму чисел соответствующей подпоследовательности найдём по формуле $i*(i + 1)/2$. А на частичных подпоследовательностях сумму найдём перебором. Асимптотика $O(R^{1/2})$.

Задача 2. Дружелюбные соседи

Вспомогательная задача

По условию нам дан циклический массив натуральных чисел. Прежде, чем приступить к решению оригинальной задачи, решим аналогичную задачу для обычного массива.

Если мы решаем задачу для обычного массива, то первая перегородка определена и находится перед первым элементом массива. Найдём вторую перегородку первой комнаты. Для этого воспользуемся жадным алгоритмом. Жадность заключается в следующем: первый марсианин обязан находится в первой комнате, добавим туда второго, если он не враждует с первым, потом добавим третьего, если он не враждует с первым и вторым и т. д. Таким образом добавим максимальное возможное количество марсиан в первую комнату. Аналогично находится положение остальных перегородок. Итого, если для каждого из **N** марсианинов за **O(N)** проверять, сможем ли мы добавить его в комнату, то получаем асимптотику $O(N^2)$.

20 баллов ($N \le 50$, $A_i \le 2$)

Перейдём к оригинальной задаче. Сведём решение оригинальной задачи к решению вспомогательной. Чтобы данную задачу решать так же, как вспомогательную, надо зафиксировать одну перегородку. Тогда если мы “разрежем” циклический массив в этой перегородке, то получим обычный массив, а следовательно сведём задачу к предыдущей. На перебор перегородки потратим $O(n)$, а на решение оригинальной задачи для каждой перегородки уйдёт $O(n^2)$. Тогда итоговая асимптотика $O(n^3)$.

Здесь и далее дружелюбность марсиан будем записывать в двоичной форме. Тогда в данной подзадаче есть всего два вида дружелюбности 01_2 и 10_2. Заметим, что $01_2 \,\&\, 10_2 = 00_2$, а $01_2 \,\&\, 01_2 = 01_2 \neq 0$ и $10_2 \,\&\, 10_2 = 10_2 \neq 0$. Тогда в данной подзадаче марсиане не дружат друг с другом тогда и только тогда, когда их дружелюбности совпадают. Это позволяет не использовать битовые операции в решении.

40 баллов ($N \le 50$, A_i - степень двойки)

Так как A_i - степень двойки, то A_i в имеет вид $1_2, 10_2, 100_2, 1000_2, \ldots$

Тогда, по аналогии с предыдущей подзадачей, марсиане не дружат друг с другом тогда и только тогда, когда их дружелюбности совпадают. Нетрудно видеть, что решение предыдущей группы будет работать и в этой. Асимптотика $O(n^3)$.

60-80 баллов ($N \le 50$, $N \le 300$)

Заметим, что если в решении использовать битовые операции для проверки $A\&B \neq 0$, то можно использовать всё то же решение для предыдущих групп. Асимптотика $O(n^3)$. Здесь на 60 баллов, как и в предыдущих группах, допускается менее оптимальная реализация.

90 баллов ($N \le 2000$)

Сделаем следующее ключевое наблюдение: $A \,\&\, B \neq 0$ тогда и только тогда, когда существует разряд i в двоичной записи числа такой, что $A[i] = B[i] = 1$, где $A[i], B[i]$ являются цифрами, стоящими в соответствующих разрядах i.

Исходя из этого наблюдения получаем, что марсианин с дружелюбностью A_i не будет враждовать с уже имеющимися в комнате марсианами, когда в комнате нет такого марсианина, дружелюбность которого имеет общий единичный разряд с A_i. Давайте поддерживать массив(маску) C встречавшихся в комнате единичных разрядов. Тогда для того, чтобы проверить, что A_i не враждует с уже имеющимися в комнате марсианами, достаточно проверить, что A_i не имеет общих с C единичных разрядов. Учитывая это, вспомогательную задачу можно решить за $O(n)$, а значит всю за $O(n^2)$.

100 баллов

Для полного решения надо заметить ещё один ключевой факт: количество марсиан в каждой комнате не превышает **32**.

Докажем. Это напрямую следует из предыдущего факта, так как $A_i \neq 0$ и не имеет общих единичных битов с **C**, то в A_i есть хотя бы один единичный бит которого нет в **C**. Следовательно, после добавления A_i в комнату в **C** появится хотя бы один новый единичный разряд. Из $A_i \leq 10^9$ следует, что различных единичных разрядов в **C** может быть не более **32**. Следовательно, добавлений A_i может быть не более **32**.

Тогда для первой перегородки достаточно перебрать только первые $w = 32$ позиции, для каждой мы будем решать подзадачу за **O(n)**. Асимптотика **O(w*n)**, где **w** - количество бит A_i.

Задача 3. Поворот плиток.

20-40 баллов (N, M ≤ 10, K = 1, K = 2)

Очевидное решение. Переберём плитки, которые будем поворачивать, и для каждого поворота посчитаем количество улиц, которое получается после поворота. Выберем среди них минимальное.

60 баллов (N, M, K ≤ 50)

Достаточно было написать жадность, которая перебирает плитки и поворачивает их только тогда, когда это улучшает ответ.

100 баллов

Разделим все улицы на два типа: замкнутые (циклы) и незамкнутые (цепочки).

Сделаем несколько замечаний:

- Если два цикла имеют общую плитку, то при повороте этой плитки данные два цикла объединятся, и мы получим один цикл.
- Если цикл и цепочка имеют общую плитку, то при повороте этой плитки цикл и цепочка объединятся, и мы получим одну цепочку.
- Если две цепочки имеют общую плитку, то при повороте этой плитки мы снова получим две цепочки.

100 баллов

- Каждая плитка либо принадлежит ровно одной улице, либо является общей для некоторых двух улиц и только для них.
- При повороте плитки, которая не является общей для некоторых двух улиц, количество улиц не уменьшается.
- Общая для двух циклов или для цикла и цепочки плитка после поворота будет принадлежать ровно одной улице.

Из этих замечаний следует, что есть смысл поворачивать только те плитки, которые являются общими для двух циклов или для цикла и цепочки.

100 баллов

Из данных замечаний нетрудно видеть, что будет работать следующая жадность: Пока есть плитки, общие для двух циклов, то поворачиваем их, Если таких нет, но есть плитки, общие для цикла и цепочки, то поворачиваем их. Если их также нет, то ответ улучшить нельзя. После каждого поворота количество улиц строго уменьшается на **1**.

Плитки, общие для двух циклов и для цикла и цепочки, можно найти заранее при помощи **BFS** или **DFS**. Обратите внимание, что при повороте общей для цикла и цепочки плитки, надо проверить, что данный цикл уже не превратился в цепочку после некоторого поворота!

Задача 4. Садовник

Интересный факт

Для начала заметим один интересный факт: если изначально в массиве X различных значений, то выполняя операции, описанные в условии задачи, после каждого запроса в массиве будет не более X различных значений. Причём, если элементы на позициях i и j были равны, после каждого запроса они останутся равны.

Действительно, если пришёл запрос обновления, то ко всем значениям x_i прибавилась 1. А значит после запроса в массиве не осталось значений x_i, но могли появится новые значения $(x_i + 1)$ на соответствующих позициях. А значит, количество различных значений не увеличилось, и все значения на соответствующих позициях остались равны.

30 баллов ($N, Q \le 100\,000, Z_i \le 50$)

Т.к. $Z_i \le 50$, то в массиве изначально не более **50** различных значений, а из “интересного факта” следует, что их количество при выполнении запросов не увеличится и соответствующие элементы будут оставаться равны. Заведём для каждого различного элемента x_i массив префиксных сумм $pref_i[j]$, в котором суммой на префиксе будет количество элементов равных x_i на префиксе **j**.

Запрос 1: Просматриваем все различные x_i и если он соответствует запросу, то прибавим к нему 1. При этом соответствующий ему массив префиксных сумм не изменится, так как соответствующие элементы стали равны $(x_i + 1)$.

30 баллов ($N, Q \le 100\,000, Z_i \le 50$)

Запрос 2: Просматриваем все различные x_i. Если $x_i \le y_j$, то прибавляем к ответу сумму на отрезке $pref_i[R_j] - pref_i[L_j - 1]$.

Тогда на каждый из $O(Q)$ запросов тратится $O(max(Z_i))$. Предпросчёт $O(max(Z_i))$ массивов $pref_i[j]$ происходит за $O(N)$ каждый. Итоговая асимптотика $O(max(Z_i)*(N + Q))$.

Оптимизации в offline задаче

Заметим, что нам совершенно необязательно отвечать на запросы по ходу чтения их из файла, а можно сразу прочитать все запросы и обработать их вместе (так называемая **offline** постановка задачи).

Заметим, что ячейка X_i не включена в ответ запроса второго типа **[L,R,Y]** тогда и только тогда, когда существует последовательность запросов первого типа X_i, X_{i+1}, ... **Y** на позициях, предшествующих **[L, R, Y]**.

Поддерживая массив “использованных” значений X_i в запросах первого типа, за время **O(N)** можно свести оригинальную постановку к подзадаче, содержащей только запросы второго типа **[L, R, Y_n]** на оригинальной последовательности.

60 баллов ($N, Q \le 200\,000, Z_i \le 200$)

Произведем предварительную подготовку для **offline** задачи. Для каждого запроса второго типа (напомним, что запросов первого типа более не осталось) найдем необходимые массивы префиксных сумм, в которых надо взять $\mathbf{pref_i[R_j] - pref_i[L_j - 1]}$. На основании этого для каждого $\mathbf{x_i}$ составим список запросов. Далее для каждого $\mathbf{x_i}$ поочерёдно строим массивы префиксных сумм и отвечаем на весь список запросов. Асимптотика такая же $\mathbf{O(max(Z_i)*(N + Q))}$.

90 баллов ($N, Q \le 200\,000, Z_i \le 100\,000$)

Доработаем решение на **60** до решения на **90**. Для этого заметим, что могут быть такие x_i, которым соответствует достаточно мало позиций в массиве и строить для такого x_i массив префиксных сумм не выгодно. Так, например, x_i, которым соответствует не менее $N^{1/2}$ позиций, не более чем $N^{1/2}$, такие x_i назовём толстыми. В то время как x_i, которым соответствует менее $N^{1/2}$ может быть достаточно много, такие x_i назовём тонкими.

Тогда в решении на **60** будем строить массив префиксных сумм только для толстых x_i. Для тонких x_i просто сохраним множество позиций, на которых они находятся.

Таким образом мы добились асимптотики $O(N^{1/2} * (N + Q))$.

100 баллов

А теперь решение на **60** доработаем на **100** баллов.

Вместо массива префиксных сумм, у нас теперь будет дерево отрезков (Фенвика), которое нам понадобится, чтобы постоянно не пересчитывать весь массив префиксных сумм. Дополнительно упорядочим x_i по возрастанию. Тогда мы можем построить дерево для x_j из дерева для $x_i < x_j$, добавив единицу ко всем позициям, где встречаются x_j. Заметим, что каждую позицию мы добавляем в дерево ровно один раз. Так как дерево Фенвика позволяет выполнять эти запросы за **O(log(N))**, получаем итоговую асимптотику решения **O(N*log(N))**.

Typ 2

Задача 1. Доставка почты

96 баллов (N ≠ 1, M ≠ 1)

Давайте раскрасим таблицу в шахматном порядке. Тогда нетрудно видеть, что почтальон будет ходить по таблице так же,, как слон на шахматной доске.

Если почтальон стоит в чёрной клетке, то он сможет посетить все чёрные клетки.

Если почтальон стоит в белой клетке, то он сможет посетить все белые клетки.

Тогда задача сводится к тому, чтобы определить цвет клетки в шахматной раскраске.

Нетрудно видеть, что для этого чётность суммы координат почтальона $(X_0 + Y_0)$ совпадает с чётностью суммы координат адреса $(X_i + Y_i)$.

100 баллов

Здесь можно было написать любой обход **(BFS, DFS).**

В чём же проблема предыдущего решения?

Проблема с тестами, где **N = 1** или M = 1. В этом случае почтальон никуда не может сдвинуться независимо от четности суммы координат точки назначения. Следовательно, доставка возможна исключительно в точку **(X_0, Y_0)**

Задача 2. Контрольная работа

20 баллов ($N \le 100$, $M = K = 2$)

В этой группе тестов в массиве может быть всего два разных значения **1** и **2**. Тогда мы можем получить 2^N различных массивов длины **N**, состоящих из **1** и **2**.

Заметим, что в этом случае единственным массивом, в котором произведение чисел не делится на **2**, является единичный массив **(1, 1, 1, ... , 1, 1)**.

Тогда ответом на данную группу будет $2^N - 1$.

40 баллов ($N \le 100$, $M, K \le 3$)

Разберём случаи:

1) $M \le K$. Тогда среди чисел массива должно быть хотя бы одно число M, так как все остальные взаимно просты с M. Найдём количество массивов, где есть хотя бы одно число M. Для этого от числа всех возможных массивов вычтем количество тех, где которых нет числа M. Ответ в этом случае будет $K^N - (K - 1)^N$.
2) $M > K$. Очевидно, что решений нет. Ответ 0.

60 баллов (N, M, K ≤ 500)

В данной группе можно написать следующую динамику:

dp[i][j] - количество способов получить массив длины **i**, и **НОД(P, M) = j**, где **P** - произведение элементов массива.

В данной динамике будут следующие переходы:

dp[i][j] ⟶ dp[i + 1][НОД(j*k, M)], где **k** является некоторым числом, которое мы хотим дописать к массиву. Ответ будет содержаться в **dp[N][M].**

Асимптотика решения **O(N*M*K)**.

80 баллов (N, M, K ≤ 2000)

Подметим тот факт, что **dp[i][j]** принимает ненулевые значения только для тех **j**, которые являются делителями **M**. Таких в каждом слое динамики будет не более $M^{1/2}$ штук.

Предварительная подготовка за **O(M)** позволяет снизить итоговую сложность решения до $O(N*M^{1/2}*K)$

100 баллов

Помимо сокращения множества рассматриваемых **j**, можно сократить множество рассматриваемых переходов в **j**. Заметим, что в с

остояние **dp[i][j]** можно перейти только из состояния **dp[i - 1][g]**, где **g** является делителем числа **j**. Тогда заранее рассчитаем переходы между слоями, запомнив значения **НОД**. Суммарное число переходов между слоями для ограничений задачи, таким образом, не превзойдет **900.**

Сложность итогового решения - **O(900 N)** простых операций.

Задача 3. Текстовый редактор

Расчет непохожести

Нетрудно заметить, что непохожесть двух строк **A** и **B** равняется **|A|+|B| - 2 lcp(A, B)**, где **|S|** - длина строки, **lcp(U, W)** - длина наибольшего общего префикса (longest common prefix) строк **U**, **W**.

40 баллов ($N \le 100$)

Рассчитываем $\mathbf{lcp(S_i, S_j)}$ для всех возможных пар строк, после чего рассчитываем уникальность множества строк “в лоб” согласно определению из условия задачи.

Итоговая сложность - $\mathbf{O(LN + N^2)}$, где $\mathbf{L}$ - суммарная длина строк. Первый член в выражении возникает следующим образом - нам необходимо произвести $\mathbf{N^2}$ расчетов lcp, занимающих время $\mathbf{min(|A|, |B|)}$ для пары строк $\mathbf{A}$, $\mathbf{B}$. Данное выражение максимизируется, если положить все строки равными и имеющими длину $\mathbf{L/N}$.

60 баллов ($N \le 5000$)

Отсортируем строки лексикографически, и рассчитаем длины наибольших общих префиксов для соседних строк - $lcp(S_i, S_{i+1})$. Такую предварительную подготовку можно осуществить за время $O(L \log N + L)$.

Для произвольной пары строк $lcp(Si, Sj)$ можно вычислить из просчитанных значений: $lcp(S_i, S_j) = \min\{ lcp(S_i, S_{i+1}), lcp(S_{i+1}, S_{i+2}), ..., lcp(S_{j-1}, S_j) \}$

Таким образом, ответ задачи может быть вычислен за итоговое время $O(L \log N + L + N^2)$

100 баллов

Прошлое решение является неполным по двум причинам - медленной сортировке на шаге предварительной обработки и квадратичной сложности расчета уникальности.

Ускорить сортировку можно, применив любой из специализированных строковых алгоритмов - либо построив суффиксный массив от конкатенации строк за время $O(L \log L)$ (в худшем случае), либо применив любой вариант быстрой сортировки для строк (например, алгоритм Бентли-Седжвика).

100 баллов

Для того, чтобы избавиться от квадратичной сложности на втором шаге, подметим следующий факт - последовательность $lcp(S_i, S_j)$ неубывает с ростом i для фиксированного j. Разобьем эту последовательность на группы одинаковых значений, поместим самую левую позицию каждой группы в стек. Тогда при переходе от j к $j+1$ можно удалить из стека все значения, большие $lcp(S_j, S_{j+1})$, чтобы сохранить неубывание. Суммарная сложность этого пересчета $O(N)$, амортизированная сложность каждого шага $O(1)$. Нетрудно заметить, что сумму значений на префиксе можно поддерживать при этом пересчете.

Задача 4. Послание инопланетянам

Наименьшая общая надстрока

Нетрудно заметить, что условие задачи является практически незавуалированной формулировкой известной задачи о **наименьшей общей надстроке** (minimum superstring problem). Данная задача имеет большое прикладное значение, в частности, в медицине и биологии для секвенирования ДНК. К сожалению, данная задача является **NP-трудной**. В простых терминах это означает, что оптимального решения с полиномиальной сложностью для нее скорее всего не существует. Поэтому от участников олимпиады ожидаются экспоненциальные решения (переборы) с различными отсечениями. Стоит отметить, что значительная часть тестов содержит небольшое число строк (менее 15), что позволяет найти оптимальное решение за разумное время (до десятков минут) в ходе соревнования. Некоторые тесты можно решить вручную.

Сведение к задаче о коммивояжере

Рассмотрим пример такой эвристики. Для начала удалим из множества все строки, являющиеся подстроками других строк множества. Тогда мы можем сопоставить каждой строке вершину ориентированного графа, а каждой паре строк - ребро веса, равного длине максимального суффикса первой строки, являющегося префиксом второй. Тогда для решения задачи нам будет необходимо решить задачу о поиске пути максимального веса, посещающего все вершины графа ровно один раз (задачу о коммивояжере).

Такую задачу можно решить “в лоб” за время $O(N!)$, либо при помощи динамического программирования за время $O(N*2^N)$