All-Ukrainian Collegiate Programming Contest 2022 I stage Розбір задач

Дякуємо за участь!

,

Задача A. Абсолютно неадекватні слова

Для кожного рядка, просто перевірте, чи є в ньому символи `V` чи `Z`. Це можна зробити за O(довжина рядка).

,

Задача B. Мінус сума

Розглянемо число $x = -(a_1, a_2, \ldots, a_n)$. Помітимо, що після операції з індексом i, a_i стає рівним x, а x стає рівним $-(a_1 + a_2 + \ldots + a_{i-1} + (-(a_1 + a_2 + \ldots + a_n)) + a_{i+1} + a_{i+2} + \ldots + a_n) = -(-a_i) = a_i$. Іншими словами, наша операція просто міняє місцями числа a_i та x.

Тепер ми можемо переформулювати задачу так: є масив з n чисел, та число x. За одну операцію ми можемо вибрати довільне i від 1 до n, і поміняти місцями a_i та x. Скільки різних масивів a ми можемо отримати?

Неважко показати, що такими операціями ми можемо обміняти взагалі будь-які два елементи. Наприклад, щоб обміняти місцями a_i, a_j, досить зробити наступну послідовність операцій: міняємо $(a_i, x), (a_i, a_j), (x, a_j)$ (тут в кожній парі другим йде число, що зараз знаходиться в цій позиції "поза масивом"). Отже, ми можемо переставити всі ці $n + 1$ чисел взагалі будь-яким чином.

Масив a визначає і число x, тому нам достатньо дізнатись кількість цих різних перестановок цих $n + 1$ числа. Легко побачити, що кількість різних перестановок чисел $b_1, b_2, \ldots, b_m$ рівна $\frac{m!}{\prod_y cnt(y)!}$, де $cnt(y)$ — це кількість входжень x в масив. Дійсно, ми можемо переставити m елементів довільним чином, але маємо поділити на кількості способів переставити між собою входження однакових елементів.

Отже, в оригінальній задачі, потрібно порахувати кількості входжень кожного числа y в масив $a \cup$ число x — $cnt(y)$, і знайти $\frac{(n+1)!}{\prod_y cnt(y)!}$.

Нагадаємо, що ми можемо знаходити факторіали та обернені факторіали всіх чисел до n за простим модулем p за $O(n + \log p)$: спочатку знаходимо значення $i!$ для $1 \leqslant i \leqslant n$, потім знаходимо $\frac{1}{n!}$ за $O(\log p)$, знайшовши $(n!)^{p-2} \bmod p$, а потім знаходячи $i!$ як $(i + 1) \cdot \frac{1}{(i+1)!}$ для $1 \leqslant i \leqslant n - 1$.

Задача C. Найближчі точки

Покажемо, як знаходити найближчу точку для конкретної точки X. Ми будемо знаходити найменший квадрат відстані від X до інших точок, і лише потім брати корінь, щоб всі порівняння були в цілих числах.

Якщо $X = (0, 0)$, то просто переберемо всі інші точки за $O(n)$. Інакше, лише одна координата X рівна 0. Розберемо випадок, коли це x-координата, тобто точка X лежить на осі Oy, і має вигляд $(0, a)$.

Яка точка на осі Ox найближча до X? Квадрат відстані від точки $(0, a)$ до точки $(b, 0)$ рівний $a^2 + b^2$, тобто нам просто потрібна точка $(b, 0)$ з осі Ox з найменшим значенням $|b|$.

Яка точка на осі Oy, відмінна від X, найближча до X? Очевидно, точка з найближчою y-координатою.

Отже, можна зробити наступне: знайдемо мінімальне значення $|b|$ по всім точкам $(b, 0)$ з осі Ox, а також відсортуємо всі точки з осі Oy (крім $(0, 0)$) за y координатою. Для кожної точки з цієї осі, візьмемо мінімум з трьох відстаней: від неї до найближчої точки з Oy зверху, від неї до найближчої точки з Oy знизу, а також від неї до найближчої точки з Ox (цю відстань ми вміємо рахувати за $O(1)$, якщо попередньо знайдемо мінімальне значення $|b|$ по точкам $(b, 0)$ з осі Ox).

Аналогічно зробимо для точок з осі Ox. Загальна асимптотика $O(n \log n)$ для сортування точок.

Не забудьте, що може бути й таке, що на осі Ox не буде жодної точки.

Задача D. Запити

Помітимо, що порядок чисел не змінюється, якщо до всіх чисел додати одне й те саме число. Це наштовхує на наступний розв'язок.

Спочатку відсортуємо всі числа, отримавши масив $b_1 \geqslant b_2 \geqslant \ldots \geqslant b_n$. Також, будемо тримати змінну $adder$, що позначає, скільки ми додали до всіх елементів масиву.

Тоді у відповідь на запит першого типу, ми просто додаємо x до $adder$. У відповідь на запит другого типу, ми виводимо $b_k + adder$.

Асимптотика $O(n \log n + q)$.

Задача E. Закусити

Спершу поглянемо, як для даної перестановки визначати її красу.

Якщо вся перестановка непарна, то її краса рівна n. Припустимо, що перестановка парна.

Для i позначатимемо за c_i кількість інверсій, в які входить p_i, тобто число індексів j, для яких $j < i$ та $p_j > p_i$, або $j > i$ та $p_j < p_i$. Якщо якесь з чисел c_i непарне, то можна розглянути підпослідовність з всіх елементів, крім i-го, і вона буде непарною. В цьому випадку краса буде $n-1$.

Інакше, всі c_i теж парні. Припустимо, що в перестановці є хоч одна інверсія, скажімо, $i < j$ та $p_i > p_j$. Тоді розглянемо підпослідовність, що містить всі елементи, крім i та j. Вона непарна (адже ми видалили всі інверсії, що містять p_i чи p_j, але інверсію, утворену p_i та p_j ми маємо видалити лише один раз). Отже, в цьому випадку краса $n-2$. Якщо ж інверсій немає, то перестановка рівна $(1, 2, \ldots, n)$. Всі її підпослідовності відсортовані, тому її краса -1.

Лишилось зрозуміти, як швидко знаходити цю красу. Помітимо наступне: c_i парне тоді й тільки тоді, коли $p_i \bmod 2 = i \bmod 2$. Покажемо це. Нехай x — кількість індексів j від 1 до $i-1$, для яких $p_j < p_i$. Тоді зліва від p_i $i-1-x$ елементів утворюють з a_i інверсію. Справа ж $p_i - 1 - x$ елементів утворюють з a_i інверсію. Отже, всього a_i лежить в $i-1-x+p_i-1-x = (i+p_i)-2(x+1)$ інверсіях. Отже, c_i парне тоді й тільки тоді, коли $i+p_i$ парне, тобто лише коли i та p_i мають однакову парність.

Порахуємо парність p з самого початку. Нескладно помітити, що парність p рівна парності числа $n-cycles$, де $cycles$ — кількість циклів в перестановці p. Дійсно, обмін місцями двох елементів змінює парність перестановки (дуже відомий факт), а ми можемо відсортувати нашу перестановку рівно за $n - cycles$ обмінів місцями двох елементів.

Також, будемо тримати cnt_eq, cnt_same_par — кількість індексів, для яких $p_i = i$, та кількість індексів, для яких $p_i \bmod 2 = i \bmod 2$. Зрозуміло, як оновлювати ці величини після кожного обміну.

Тепер, відповідатимемо на кожен запит наступним чином: якщо перестановка непарна, краса n. Інакше, якщо $cnt\_same\_par \neq n$, краса $n-1$. Інакше, якщо $cnt\_eq \neq n$, то краса $n-2$, а інакше перестановка відсортована та краса -1.

Асимптотика $O(n+q)$.

Задача F. Універсальний обмінювач

Подивимось на позицію, де знаходиться число n. Ми маємо вміти пересувати n в будь-яку позицію. В той же час, ми можемо посунути n з позиції u в позицію v (за одну операцію) тоді й лише тоді, коли $p_u > p_v$.

Отже, якщо ми можемо пересунути n з будь-якої вершинки в будь-яку, то в орієнтованому графі, ребрами якого є (u_i, v_i), з будь-якої вершинки можна дістатись до будь-якої (такі графи також називаються **сильнозв'язними**).

Виявляється, що це достатня умова. Перед тим, як доводити це, скажемо, як визначати, що граф сильнозв'язний. Для цього досить перевірити, що можна дістатись з вершини 1 в будь-яку іншу вершину, і з будь-якої іншої вершини в вершину 1. Кожна з цих перевірок робиться одним dfsом, а тому асимптотика $O(n+m)$.

Тепер доведемо, що це достатня умова.

Розглянемо довільний орієнтований цикл в нашому графі, тобто послідовність $v_1, v_2, \ldots, v_k$ таку, що в нас є операції $(v_1, v_2), (v_2, v_3), \ldots, (v_{k-1}, v_k), (v_k, v_1)$. Покажемо, що в цьому циклі ми можемо переставляти елементи як завгодно. Без обмеження загальності, числа в позиціях $v_1, v_2, \ldots, v_k$ це $1, 2, \ldots, k$. Покажемо, що ми можемо отримати будь-яку перестановку чисел від 1 до k.

Спершу покажемо, що з будь-якої ситуації ми можемо прийти в перестановку, де $p_{v_1} = 1, p_{v_2} = 2, \ldots, p_{v_k} = k$. Дійсно, спершу рухаємо k по колу, поки воно не опиниться в v_k, потім $k-1, \ldots$, потім 2 і 1.

Тепер покажемо, що з перестановки $(1, 2, \ldots, k)$ можна отримати якийсь циклічий зсув перестановки $(k, k-1, \ldots, 1)$. Покажемо це за індукцією, для $k = 1, 2$ твердження очевидне. Перехід: спершу обміняємо k з наступним по циклу числом $k-2$ рази, отримавши перестановку $(2, 3, \ldots, k-2, k, k-1, 1)$. Тепер "об'єднаємо в один елемент"k і $k-1$ (ми можемо обмінювати їх разом: $(k, k-1, x) \to (k, x, k-1) \to (x, k, k-1)$). Маємо твердження для $k-1$.

Тепер покажемо, що з зсуву $(k, k-1, \ldots, 1)$ можна отримати будь-яку іншу перестановку. Давайте розвернемо всі операції: якщо оригінальна операція була (u_i, v_i), розглядатимемо операцію (v_i, u_i). Досить показати, що, застосовуючи ці операції, можна з будь-якої перестановки прийти в циклічний зсув $(k, k-1, \ldots, 1)$, але це ми робимо повністю аналогічно: спочатку ставимо k на місце, потім $k-1, \ldots, 2, 1$.

Це завершує доведення твердження для циклу. Далі легко показати, що ми можемо отримати будь-яку перестановку, якщо граф сильнозв'язний. Наприклад, так: розглянемо довільні дві позиції u, v $(u \neq v)$. Існує простий цикл, в якому є позиції u та v. В цьому циклі ми можемо переставити елементи довільним чином, тому ми можемо й просто обміняти місцями p_u, p_v. Отже, ми можемо обміняти місцями довільні два елементи, тому ми можемо отримати будь-яку перестановку з будь-якої.

Задача G. Мінімізуйте суму

Помітимо, що XOR всіх чисел в таблиці рівний XOR всіх row_i, а також рівний XOR всіх col_i. Отже, якщо XOR всіх row_i не рівний XOR всіх col_i, то такої таблиці не існує.

Виявляється, у всіх інших випадках вона існує. Побудуємо її по кожному біту окремо. Нехай з чисел row_i x містять біт 2^t, а з чисел col_i y містять біт 2^t. Очевидно тоді, що в таблиці принаймні $max(x, y)$ чисел містять біт 2^t, отже, ми повинні додати до відповіді принаймні $max(x, y)2^t$.

Покажемо, що існує таблиця, в якій рівно $max(x, y)$ чисел містять біт 2^t. З того, що XOR всіх row_i не рівний XOR всіх col_i, маємо, що $x \bmod 2 = y \bmod 2$.

Без обмеження загальності, $x \leqslant y$, і біт 2^t присутній в числах $row_1, row_2, \ldots, row_x, col_1, col_2, \ldots, col_y$. Тоді поставимо біт 2^t в числа в клітинках $(1, 1), (2, 2), \ldots, (x, x)$, а також $(1, x + 1), (1, x + 2), \ldots, (1, y)$. В кожному з перших y стовпчиків ми вибрали рівно одну клітину, в першому рядку ми вибрали $y - x + 1$ клітину, а в інших з перших x рядків також по одній, отже XORи будуть, як потрібно.

Отже, потрібно просто пройтись по всім 2^t, і додати до відповіді $max(x, y)2^t$. Асимптотика $O((m + n) \cdot 20)$.

Задача H. Фарбування кола

Розв'язок 1

Спершу розберемо випадок, коли всі b_i рівні. Без обмеження загальності, $b_i = 3$ для всіх i, тобто всі a_i мають дорівнювати 1 чи 2. Оскільки сусідні a_i мають бути різними, то вони мають чергуватись. Очевидно, це чергування можливе, якщо n парне, і неможливе, якщо n непарне.

Тепер розглянемо варіант, коли не всі b_i рівні. Без обмеження загальності, $b_1 \neq b_2$. Тоді давайте пофарбуємо точку 2 в колір b_1, а далі по черзі фарбуватимемо інші точки, в порядку від 3-ої до n-ої, а потім 1-шу, в будь-який дозволений колір. Коли ми фарбуємо точку i, де $3 \leqslant i \leqslant n$, ми не можемо пофарбувати її лише в кольори b_i та a_{i-1}, тому хоч один колір лишається дозволеним. Коли ми фарбуємо точку 1, то заборонені лише кольори a_n та $b_1 = a_2$, тому ми також можемо її пофарбувати.

Асимптотика $O(n)$.

Розв'язок 2

Також можна розв'язати цю задачу без розбору випадків, суто динамічним програмуванням.

Нехай $dp[first\_color][pos][color] = true$, якщо можна розфарбувати перші pos точок правильним чином, щоб колір першої точки був $first\_color$, а точки pos — $color$. Рахуватимемо це для всіх $first\_color, color$, для pos від 1 до n. Коли ми вже порахували ці значення для $pos - 1$, то для pos їх обчислити просто: $dp[first\_color][pos][color] = true$ лише якщо $color \neq b_{pos}$, і існує колір $color_1 \neq color$, для якого $dp[first\_color][pos - 1][color_1] = true$ (а також, у випадку $pos = n$, має справджуватись $first\_color \neq color$. Якщо такий $color_1$ існує, запам'ятаємо його.

Розфарбування можливе лише в тому разі, якщо $dp[first\_color][n][color] = true$ для якихось $first\_color$ та $color$. Якщо воно можливе, неважко відновити його з допомогою збережених $color_1$.

Асимптотика $O(3^3 \cdot n) = O(n)$.

Задача I. Проста умова

Для зручності, будемо нумерувати вершини від 0 до $n-1$.

Спершу, будемо позначати підмножину вершин з вершин $i_1, i_2, \ldots, i_k$ як $2^{i_1}+2^{i_2}+\ldots+2^{i_k}$. Таким чином, кожній підмножині вершин графу відповідає якесь число $mask$ від 0 до 2^n-1.

Розв'язок за $O(2^n n^3)$.

Нехай $dp[mask][u][v]$ позначає, чи існує для графу, на вершинах з підмножини $mask$ Гамільтонів шлях, що починається в вершині u, а закінчується в вершині v (таким чином, $mask$ має містити біти u та v). Ми можемо почати з того, щоб для кожної вершини v зробити $dp[2^v][v][v] = true$.

Будемо перебирати підмножини в порядку зростання їх розміру. Зверніть увагу, що $dp[mask][u][v] = true$ тоді й лише тоді, коли в $mask$ є якась вершина $w \neq v$, така, що:

- w з'єднана з v
- $dp[mask - 2^v][u][w] = true$

(Ця вершина w — це вершина, сусідня до v в цьому потенційному Гамільтоновому шляху).

Отже, для даної $mask$ та даної пари (u, v), ми можемо перебрати всі кандидати w за $O(n)$, що дає $O(2^n n^3)$ загалом.

Розв'язок за $O(2^n n^2)$.

Замість булевої змінної $dp[mask][u][v]$ давайте тримати число $dp_1[mask][u]$. В ньому біт буде присутній біт v тоді й лише тоді, якщо $dp[mask][u][v] = true$. Також, для кожної вершини u давайте тримати число $neigh_i$, в якому присутній біт v тоді й лише тоді, коли між вершинами u та v є ребро.

Знову ж, давайте рахувати значення $dp_1[mask][u]$, перебираючи $mask$ в порядку зростання розміру підмножин. Для кожної вершини v, що є в $mask$, спробуємо визначити, чи є біт v в $dp_1[mask][u]$. Згадаємо, що $dp[mask][u][v] = true$ тоді й лише тоді, коли в $mask$ є якась вершина $w \neq v$, така, що: w з'єднана з v та $dp[mask - 2^v][u][w] = true$. Це рівносильно тому, що в $dp_1[mask - 2^v][u]$ та $neigh_v$ є принаймні один спільний біт! Ми можемо з'ясувати, чи є в них спільний біт, за одну операцію `AND`. Отже, для даних $mask, u$ ми можемо порахувати $dp_1[mask][u]$ за $O(n)$, а тому загальна асимптотика $O(2^n n^2)$.

Розв'язок за $O(2^n n)$.

Цього все ще не достатньо, необхідно зробити ще якусь оптимізацію.

Давайте порахуємо значення $dp_1[mask][n-1]$ для всіх $mask$. Ми робимо це за $O(2^n n)$. Помітимо, що кожен Гамільтонів шлях оригінального графу має проходити через вершину $n-1$. З обчислених значень dp_1 легко зрозуміти, до яких вершин є Гамільтонів шлях від вершини 1. Розглянемо якісь інші дві вершини $u, v < n-1$. Легко бачити, що між вершинами u та v існує Гамільтонів шлях тоді й лише тоді, коли існують дві підмножини $mask_u, mask_v$ такі, що:

- $u \in mask_u, v \in mask_v$
- Кожна вершина, крім $n-1$, знаходиться рівно в одній з цих підмножин. Вершина $n-1$ знаходиться в обох підмножинах.
- $u \in dp_1[mask_u][n-1], v \in dp_1[mask_v][n-1]$

Інакше кажучи, лише тоді, коли існує розбиття вершин на 2 групи (з $n-1$ в кожній групі), що в першій групі існує Гамільтонів шлях від u до $n-1$, а в другій від $n-1$ до v.

Але як швидко визначити це для кожної пари вершин? Для кожної вершини u давайте тримати число ans_u, біт v в якому присутній тоді й лише тоді, коли в графі існує Гамільтонів шлях від u до v. Тепер, давайте переберемо всі $mask_u$ ($mask_v$ однозначно визначається за $mask_u$, як $2^n - 1 - mask_u + 2^{n-1}$), і для кожної $mask_u$ переберемо всі u. Для кожної з цих u зробимо $ans_u = ans_u$ `OR` $dp_1[mask_v][n-1]$.

Загальна асимптотика $O(2^n n)$.

Задача J. Зростаюча таблиця

Давайте віднімемо від $a_{i,j}$ число $i+j-1$. Легко бачити, що тепер всі рядки та стовпці мають бути неспадаючими. Також, $0 \leqslant a_{1,1}$ та $a_{n,m} \leqslant 1$. Отже, всі числа мають бути від 0 до 1. Це й достатня умова: якщо $0 \leqslant a_{i,j} \leqslant 1$, то в початковій таблиці це число було $i+j-1$ чи $i+j$, тобто було в проміжку від 1 до $n+m$.

Отже, нас цікавлять таблиці з одиниць та нулів, що не спадають по рядкам та стовпцям, деякі елементи яких нам дані (якщо $a_{i,j} - (i+j-1) \notin \{0,1\}$ до віднімання, то таких таблиць не існує, і відразу виводимо 0).

Для кожної одиниці, всі числа в прямокутнику справа та вниз від неї мають також бути одиницями. Для кожного нулика, всі числа в прямокутнику зліва та вверх від нього мають також бути нуликами. Отже, якщо якийсь нулик знаходиться вправо та вниз від якоїсь одиниці, то відповідь відразу 0.

Ми можемо зробити заповнення вище наступним чином: для кожного рядку від 1 до n, для кожного стовпця від 1 до m, якщо $a_{i,j} = 1$, то зробимо $a_{i+1,j}$ та $a_{i.j+1}$ також рівними 1 (якщо ці клітини існують). Якщо колись ми маємо зробити нулик рівним одиничці, то розв'язків немає. Потім зробимо аналогічну річ з нуликами.

Лишилось зрозуміти, як тепер рахувати кількість способів назначити інші значення.

Помітимо, що таблиці з одиниць та нулів, що не спадають по рядкам та стовпцям, мають наступний вигляд: є якийсь шлях від нижнього лівого кута до правого верхнього, що йде по сторонами сітки лише вгору та вправо, такий, що по одну сторону від цього шляху лише нулики, а по іншу лише одиниці. Порахуємо кількість таких шляхів, якщо деякі значення ми вже знаємо.

Розглянемо вузли нашої таблиці. Вони утворені перетином $n+1$ горизонтальної та $m+1$ вертикальної лінії. Будемо позначати за $dp_{i,j}$ кількість способів провести шлях, що йде по сторонами сітки лише вгору та вправо, від нижнього лівого кута (тобто перетину n-ї горизонталі та 0-ї вертикалі) до вузла на перетині i-ї горизонталі та j-ї вертикалі, щоб всі числа зліва від цього шляху були нулями, а вниз - одиницями. Ми починаємо з $dp_{n,0} = 1$, а знайти нам потрібно $dp_{0,m}$.

Це dp досить легко порахувати: до $dp_{i,j}$ нам потрібно додати якісь з чисел $dp_{i+1,j}$ та $dp_{i,j-1}$, залежно від того, чи задовольнятиме шлях всім умовам.

Асимптотика $O(n^2)$.

Задача K. Різнобарвний кістяк

Можна було б просто побудувати граф на всіх цих ребрах, потім відсортувати їх за вагою, і вирішити задачу алгоритмом Крускала. На жаль, цих ребер може бути дуже багато (до $O(n^2)$).

Виявляється, більшість з цих ребер не важливі.

Відсортуємо вершини за значенням a_i. Тепер вважаємо, що $a_1 \leqslant a_2 \leqslant \ldots \leqslant a_n$. Ці вершини розбиваються на кілька груп послідовних вершин за кольором (в кожній групі у всіх точок однаковий колір, в точок з сусідніх груп різні кольори). Позначимо ці групи $C_1, C_2, \ldots, C_k$.

Покажемо спершу, що можна вважати, що в мінімальному кістяку присутні лише ребра між вершинами з сусідніх груп. Дійсно, розглянемо мінімальний кістяк, в якому є ребро між вершинами i та j ($i < j$) такими, що між групами i та j є принаймні одна група. Нехай i з групи C_{i_1}, а j з групи C_{j_1}. Для кожного t від $i_1 + 1$ до $j_1 - 1$, виберемо довільну точку з групи C_t, скажімо, p_t. Тепер, приберемо ребро (i, j), і розглянемо пари вершин $(i, p_{i_1+1}), (p_{i_1+1}, p_{i_1+2}), \ldots, (p_{j_1-2}, p_{j_1-1}), (p_{j_1-1}, j)$.

Після того, як ми прибрали ребро (i, j), наш кістяк розпався на дві компоненти, причому i та j лежать в різних компонентах. Отже, не може статись так, що в кожній парі вершини лежать в одній компоненті (тоді б i та j також лежали в одній компоненті). Виберемо пару, в якій вершини лежать в різних компонентах, і проведемо між ними ребро. Його вага, очевидно, не перевищує вагу ребра (i, j).

Зробимо таку процедуру для кожного ребра, що з'єднує вершини не з сусідніх груп. Тепер вважаємо, що в мінімальному кістяку присутні лише ребра між вершинами з сусідніх груп.

Тепер покажемо, що можна вважати, що кожне ребро між групами C_i та C_{i+1} містить або найправішу вершину з C_i, або найлівішу з C_{i+1} (або і те і те). Робимо аналогічно. Нехай є ребро (a, b) з $a \in C_i$, $b \in C_{i+1}$, c — найправіша вершина C_i, d — найлівіша C_{i+1}, причому $a \neq c, b \neq d$. Тоді розглянемо пари $(a, d), (c, d), (c, b)$. Після того, як ми прибираємо ребро (a, b), хоч в одній з цих пар вершини з різних компонент, тому ми можемо його додати (а вага цього ребра, очевидно, не перевищує вагу ребра (a, b).

Тепер в нас залишилось всього $O(n)$ кандидатів для ребер (для двох сусідніх груп з розмірами x та y, в нас є $x + y - 1$ кандидат на ребра між ними). Ми можемо розглянути всі ці ребра, і запустити алгоритм Крускала для них.

Асимптотика $O(n \log n)$.

Задача L. Неспадаючі масиви

Щоб масив $[b_1, b_2, \ldots, b_m]$ був **неспадаючим**, необхідно і достатньо, щоб був індекс i $(1 \leqslant i \leqslant m-1)$, для якого $b_i > b_{i+1}$. Отже, щоб розбити масив на максимальну кількість **неспадаючих** підмасивів, потрібно знайти максимальну кількість пар $(i, i+1)$, які попарно не перетинаються, для кожної з яких $a_i > a_{i+1}$.

Давайте ж йти зліва направо. Якщо ми в індексі i, і $a_i > a_{i+1}$, то ми беремо цю пару, додаємо 1 до відповіді, і продовжуємо йти, тепер з індексу $i+2$.

Асимптотика $O(n)$.