Топ секретно

Анализ

Ще означаваме с Q максималния брой зададени въпроси, необходими на даден алгоритъм. Нека с $A(i,k)$ означаваме въпроса "a_i по-малко ли е от k ?“.

Подзадача 1

Този алгоритъм възстановява изцяла пермутацията a. За всеки служител i ще задаваме въпросите $A(i,1), A(i,2), ..., A(i,N)$. Нека c е най-малката стойност, удовлетворяваща $A(i,c) = false$. $a_i = c - 1$. Ако не съществува такава стойност c, то $a_i = N$.

Така описаният алгоритъм има $Q = N^2$. Ако спрем да питаме за дадено i, щом срещнем отговор $false$, получаваме алгоритъм с $Q = \frac{(N+2)(N-1)}{2}$.

Подзадача 2

Тъй като $A(i,j_1) \leq A(i,j_2)$ за всеки $j_1 \leq j_2$, може да се използва двоично търсене за намирането на a_i. $Q = N \cdot \lceil \log_2 (N+1) \rceil$

Подзадача 3

В тази подзадача имаме допълнителното ограничение, че $a_i < a_{i+1}$ за всяко $0 \leq i \leq N-2$. Тъй като a е пермутация знаем, че $i \leq a_i \leq i+1$ за всяко $0 \leq i \leq N-1$.

Нека разгледаме каква информация ни дава отговор на въпрос от вида $A(i, i+1)$:

- Ако $A(i, i+1) = true$, то $a_i = i$ и единствената възможност за $a_0, a_1, ..., a_i$ е $0, 1, ..., i$. Следователно $a_N > i$.
- Ако $A(i, i+1) = false$, то $a_i = i+1$ и единствената възможност за $a_i, a_{i+1}, ..., a_{N-1}$ е $i+1, i+2, ..., N$. Следователно $a_N \leq i$.

Използвайки въпроси от този вид можем да използваме двоично търсене, за да намерим стойността на a_N за $Q = \lceil \log_2 (N+1) \rceil$.

Ограничението за Q за тази подзадача е по-високо, за да може и решението на подзадача 4 да я преминава.

Подзадача 4

Първо ще разгледаме втори алгоритъм с $Q = N \cdot \lceil \log_2 (N+1) \rceil$, който после ще покажем как да се оптимизира до $Q \leq 2 \cdot (N+1) + \log_2(N+1)$

Ще намерим стойността на a_N използвайки двоично търсене. Въпросът „a_N по-малко ли е от c ?“ се свежда до това да се преброят останалите стойности $a_0, a_1, ..., a_{N-1}$, които са по-малки от c и съответно $A(i,c) = true$. Нека означим броя им с $B(c)$. Тъй като в цялата редица, включвайки a_N, има точно c стойности, за които $a_i < c$, ако $B(c) = c$, то $a_N \geq c$, а ако $B(c) = c - 1$, то $a_N < c$. (винаги е вярно, че $c - 1 \leq B(c) \leq c$).

Нека сме на итерация от двоичното търсене и знаем, че $a_N \in [l, r)$ и ще проверяваме дали $a_N \overset{?}{<} c$ за $c = \lfloor \frac{l+r}{2} \rfloor$. Нека се е оказало, че $B(c) = c$, $a_N \geq c$ и следователно $a_N \in [c, r)$. Очевидно следващите стойности c', за които ще се изчислява $B(c')$, задоволяват $c' \in [c, r)$. Изчислявайки $B(c)$, ние сме открили всички индекси i, задоволяващи $A(i,c) = true$, и тъй като $c' \geq c$ можем да сме сигурни, че за тези индекси и $A(i,c') = true$. Следователно няма смисъл да задаваме повече въпроси за тях, защото можем да сме сигурни в отговора, който бихме получили.

Използвайки същите разсъждения, преди всяка итерация на двоичното търсене сме открили l индекса, такива че $A(i,c) = true\ \forall c \in [l, r)$ и $N + 1 - r$ индекса, такива че $A(i,c) = false\ \forall c \in [l, r)$. Остават само $(N+1) - l - (N+1-r) = r - l$ индекса, за които стойността на $A(i,c)$ не ни е известна и съответно трябва да питаме системата.

На стъпка i (индексирани от 0) дължината на интервала $[l, r)$ (и съответно $r - l$) е най-много $\lceil \frac{N+1}{2^i} \rceil$. Следователно:

$$Q = (N+1) + \lceil \frac{N+1}{2} \rceil + \lceil \frac{N+1}{4} \rceil + ...$$
$$Q \leq (N+1) + \frac{N+1}{2} + \frac{N+1}{4} + ... + \lceil \log_2{(N+1)} \rceil$$
$$Q \leq (N+1) \cdot (1 + \frac{1}{2} + \frac{1}{4} + ...) + \lceil \log_2{(N+1)} \rceil$$
$$Q \leq 2 \cdot (N+1) + \lceil \log_2 (N+1) \rceil$$