Resolução
Summer 2025

- Naim

(A ordem aqui é diferente da ordem no CF)

Seja PROB. (A) $t_i = r_i - l_i + 1$. Alonga! MEX

$MEX(a_{l_i} \dots a_{r_i}) \leq t_i$

Construímos a sequência:

$a = 0, 1, 2, \dots, t-1, 0, 1, 2 \dots$

Com $t = \min_{1 \leq i \leq n} t_i$

Dessa forma todos mex $= t$

e não tem como ser melhor

PROBLEMA B

SUBWAY PERSUIT

Se $k=0$ → Busca binária

$k>0$ → Após cada etapa atualizar.

$[l, r] \to [l-k, r+k]$

Note que o tamanho muda de $T \to \frac{T}{2} + 2k$. Ou ~~se~~ seja,

$T \geq 4k$.

Solução

Se $r-l+1 > 100$, quebro no meio:

Se $r-l+1 \leq 100$, chuta um random no range.

1) Swap $\to$ C entre [swaps cíclicos] adjacentes muda paridade do nº de inversões

(Array só com valores distintos)

2) Swap entre não adj. também:

$j-i$ swaps

$a_i, a_{i+1}, \dots, a_j \to a_{i+1}, a_{i+2}, \dots, a_j, a_i$

$i-j-1$ swaps

$a_j, a_{i+1}, \dots, a_{j-1}, a_i$

Ou seja, $2(i-j)-1$ swaps!

3) $i \to j \to k \to i$ = swap(i, j) + swap(j, k)

Paridade é INVARIANTE

PROB. C

Corner: se tem repetidos sempre dá

Se for par, sempre dá!

Prova: Vai consertando da esquerda p/ direita até 3 últimos:

5 1 2 3 4 → 1 2 5 3 4 → 1 2 3 4 5

5 3 4 1 2 → 1 5 4 3 2 → 1, 2, 5, 3, 4

→ 1, 2, 3, 4, 5

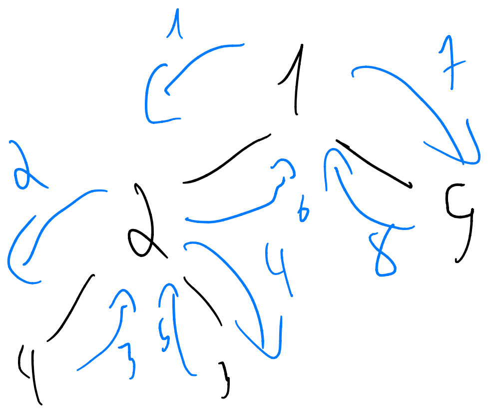
É garantido que dois últimos estão certos se paridade for par.

Vacation

→ Solução:

I) Achar MST

II) Fazer dfs e anotar in-order



Custo: $2 \cdot C_{MST}$

PROVA: A resposta ótima tem custo $\geq C_{MST}$, pois

→ Caminho Conexo

→ Visita todo nó

Ezdz

I) Se $2^k \leq a < b < 2^{k+1}$

então $2a \geq b$

II) Se agruparmos considerando bit mais significativo, todos no mesmo bloco são perigo.

III) Se $b > 2\sum_{y<b} y$, b nunca vai ter perigo

IV) Para cada bloco, olhar menor e comparar com soma de prefixo dos blocos anteriores

Edu

Update:

- Mudar soma do bloco: $O(1)$
- Manter mínimo (set)

  $O(\log N)$

F - Boxes packing

I) Podemos representar cada conjunto de caixas como uma seq. ordenada:

→ 30, 40, 50

II) Uma seq. não pode ter duas caixas de mesmo tamanho

III) Seja f_x = nº caixas de tamanho x

Resposta $\geq \max_x f_x$

Boxes

Ou seja, $\max f_x$ é um lower bound

II) É possível usar apenas

$\max_x f_x$ caixas:

→ Se precisássemos construir, basta ordenar e ir numerando as ocorrências do mesmo número p/ caixas distintas

$\overset{1}{1}, \overset{2}{1}, \overset{1}{2}, \overset{2}{2}, \overset{3}{2}, \overset{1}{3}, \overset{1}{4}, \overset{2}{4} \Rightarrow$

1	2	3
1	1	2
2	2	
3	4	
4		

Human Equation

Seja "p" array da soma de prefixos:

$$p_i = \sum_{1 \leq j \leq i} a_j = \left(\sum_{1 \leq j < i} a_j\right) + a_i = p_{i-1} + a_i$$

$p_0 = 0$, Calculado em $O(N)$.

Uma operação de $+1$ em a_i e -1 em a_j, $i < j$ é igual a somar $+1$ em $p_i, p_{i+1}, \ldots, p_{j-1}$, ou seja, intervalo $[i, j-1]$.

Human

Como queremos $p_i = 0 \ \forall i$

Fazemos:

I) Em uma 1º Fase, somamos +1 em toda posição com $p_i < 0$.

Isso é feito em $-\min(p_i)$ operações, pois em cada operação

$$p_i = \begin{cases} p_i, & \text{se } p_i \geq 0 \\ p_i + 1, & \text{se } p_i < 0 \end{cases}$$

II) Na 2º Fase, somamos -1 em toda posição com $p_i > 0$. Isso leva $\max(p_i)$ op.

TOTAL: $\mathrm{MAX}(p_i) - \mathrm{MIN}(p_i)$

Time to run

Sempre tem como andar tudo:

Até final

Se na linha $i > 1$, Até chegar na 1º coluna

Na última linha, pode subir tudo no final

volta p/ (1,1)

Magic Stones

Seja d o array de diferenças: $d_i = c_i - c_{i-1}$

Após uma operação:

$$d_i = c_i - c_{i-1} \longrightarrow d'_i = (c_{i+1} + c_{i-1} - c_i) - c_{i-1} = c_{i+1} - c_i = d_{i+1}$$

$$d_{i+1} = c_{i+1} - c_i \longrightarrow d'_{i+1} = c_{i+1} - (c_{i+1} + c_{i-1} - c_i) = c_i - c_{i-1} = d_i$$

SWAP de d_i e d_{i+1}!

Seja $dt_i = t_i - t_{i-1}$

Magic Stones

Resposta é sim see "d" = "dt" após ordenar

i.e., tem mesmos elementos

mas outra ordem.

Little Artem

Ou você pinta (quase) como xadrez, ou tudo preto menos 1 Quina:

Once in a Casino

→ Primeiro resolver sem restrição que $0 \leq a_i \leq 9$

- Da esq. p/ direita:

  Se $a_i \neq b_i$ faz $|b_i - a_i|$ operações em i e $i+1$.

  $a_i \leftarrow b_i$, $a_{i+1} \leftarrow a_{i+1} + b_i - a_i$

  Se $a_n \neq b_n$, Impossível

- Isso dá mínimo

→ Com restrição, se não posso operar em i, recursivamente tenta $i+1$.

Casino

Nesse caso temos algo como:

$0, 9, 0, 9, 0, 9, \ldots$

$b_i \geq 0$,

É garantido que eventualmente dá, pois já checamos antes com método sem restrição.

Change Free

→ Enquanto $c_i \geq 100$, uso notas

→ $c'_i = c_i \bmod 100$

I) Observação:

→ Se eu usar $x < c_i$ moedas, o troco são $(100 + x) - c_i$ moedas

→ Se $x = c_i$, 0 moedas

ou uso c_i moedas ou 1 nota

II) • Se você usar moedas, usa c_i ($-c_i$ moedas)

• Se usar nota, ganha $100 - c_i$

Change-free

Ou seja, sempre tem $-c_i$

Podemos refazer as opções como:

I) $m \leftarrow m - c_i$, custo $\emptyset$

II) $m \leftarrow m - c_i$,

$m \leftarrow m + 100$, custo $w_i \cdot (100 - c_i)$

Algoritmo (simplificado):

Para $1 \leq i \leq n$:

$m \leftarrow m - c_i$

FILA.PUSH$(w_i \cdot (100 - c_i))$;

IF $m < 0$:

custo += Fila.pop

m += 100

Desfaz operação priorizando menor custo.