

## ① Mochila

$(n) O = (n) O + (n^m) T \leq 2 \cdot nT$  se  $n \geq m$

$\begin{cases} 0 \\ OPT(i, w) \end{cases}$

, se  $i = 0$

$OPT(i, w) = \max(OPT(i-1, w), v_i + OPT(i-1, w - w_i))$ , se  $i > 0$

②  $\forall i \in [0, m], \forall w \in [0, W]$ ;  $OPT(i, w) = OPT(i-1, w)$  :  $w_i > w$

③ Se não me engano ele é pseudo polinomial..  $O(m \cdot W)$   
Mas por que pseudo?

Jogar Flappy

④ F Polinomial em função de N

⑤ V? Não há alg polinomial

⑥ F Polinomial em função da entrada, base 2

⑦ F Poly  $\rightarrow N \cdot W$

⑧ Melhor dia de compra e venda de ações:

$0 \leq x \leq y \leq m$ ;  $OPT = A[y] - A[x]$

$B[i] = A[i] - A[i-1]$ ;  $1 \leq i \leq m$

⑨ V Div Cong; Soma máxima de B;  $T(n) \leq 2T(\frac{n}{2}) + O(n) = O(n \lg n)$

$m \xrightarrow{1} \frac{m}{2} \xrightarrow{2} \frac{m}{4} \xrightarrow{3} \dots$

$\log^k(2)$

②

⑥ F Soma Máx A | Div Cong  $n T(m) \leq 2T(m/2) + O(1) = O(n)$

⑦ F Soma Máx B |  $n T(m/2) + O(1) = O(\log n)$

⑧ F  $O(n!) \parallel PD \parallel S(j) = \begin{cases} 0 & \\ \downarrow & \\ \text{Incr Máx} & \end{cases}$

⑨ F  $\text{OPT} = \max_{1 \leq j \leq m} S(j)$

⑩ F ~~⑨~~ ⑩  $2T(m/2) + O(1) = O(n)$

## ↳ Voltar Para Competir

③ Closet Pair: distância exclusiva, 2 distintos, div Cong p/ medianas 2 sub p de  $|m/2|$ , resultado final com conquer

④ F Passeia processa todos círculos  $; S2(n)$ ; Computa apenas  $O(n)$  passos.

⑤ V Divisão  $\Theta(n \log n)$  | Conquer  $\Theta(n)$  | Total  $\Theta(n \log(n))$   
É uma soma, né? Thinking:

⑥ F Conquer processa Tudo no círculo  $\parallel$  Dist exclus  $S2(n^2)$  mafiosa  
 $\parallel$  Pontos na faixa  $\Omega(n)$

⑦ F Se custo div =  $O(n)$ ; Custo conq =  $\Theta(n)$ ; Total =  $\Theta(n \log n)$   
E isso faz algum sentido?

⑧ V Conq todos no círculo / não computa ponto dist  $(P, L) > \delta$ , onde  
 $\delta = \min \text{dist}(CP(\text{left}), CP(\text{Right}))$

DS abstratos

(4) Sort in Count // dire Comp // Cont Jme // 2 listas onde  
maior de menores  $L_1 = (1, 2, \dots, m)$ ;  $L_2 = (a_1, a_2, \dots, a_m)$   
 $L_2$  = entrada do problema:

Def: Inversão:  $i \in j$  em  $L_1 \wedge (a_i > a_j)$

- a) ✓  $(1, 5, 4, 8, 10, 2, 6, 9, 3, 7)$   $\text{invo} = 17$  17
- b) f  $(10, 1, 4, 5, 8, 2, 6, 9, 3, 7)$  ~~invo~~  $\text{invo} = 17$  203
- c) ✓  $(10, 1, 4, 5, 8)$  ~~invo~~  $\text{invo} = 9$  43
- d) f  $(1, 5, 4, 8, 10, 2, 6, 9, 3, 7)$   $\text{invo} = 13$  17?
- e) ✓  $(10, 1, 4, 5, 8, 2, 6, 9, 3, 7)$   $\text{invo} = 20$  V
- ~~17~~ ~~203~~ ~~43~~ ✓
- // c)  $O(?) + O(m) + O(\log^2 ?)$

(5) n números, encontrar i maiores em ordem crescente,  
comparações. Algoritmos:

$$[a = b?]$$

- a) Ordenar lista heapSort e listar i maiores  $[n \log n]$
- b) lista de prioridades (heap), extrair máximo i vezes  $[n \log n]$
- c) Ordenar estatística  $[?]$ , encontrar i-ésimo maior, particionar a partir desse e ordenar as listas com os i maiores.
- d) ✓  $\uparrow$  é sempre melhor ou igual aos outros
- e) F por valores pequenos de  $i = O(1)$ , o de menor custo é  $\uparrow$   
[achou que seria c]
- f) V\*  $\uparrow$  é  $O(n \log m + i)$  [achou que dava pra ser]
- g) ac melhores ou iguais a a)
- h) F  $\uparrow$  ( $n + i \log n$ ) [me parece mais um  $O(n \log m + i)$   
Heap extração]

# Computando ZD

2d)

$$S(j) = \begin{cases} 0 & \text{para } j=1 \\ \max(S(j-1) + A[j] - A[j-1], 0) & \text{para } j \geq 2 \end{cases}$$

: subseq ab absolu = 2

$$\text{OPT} = \max_{1 \leq j \leq m} S(j)$$

Day	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Price	110	113	110	85	105	102	86	63	81	101	94	106	101	79	94	90	97
Change	-13	-3	28	20	-3	16	-23	18	20	-7	12	-5	-22	15	-4	7	

OPA) E o dia de compra ???

0	1	2
0	0+13	10

(f8, e0, s, 0, 1, 2, 1, 1)

Venda = 65 = 100% (f8, e2, s, 8, 2, 1, 0)

Preço sub de 100% = 100% - 35% = 65%

(f8, e0, s, 0 + 13) 0 + (13) 0

maior que 100% é menor que 100% (f8, e0, s, 0 + 13) 0 + (13) 0

contudo, algumas

(f8, e0, s, 0 + 13) 0 + (13) 0

maior que 100% é menor que 100% (f8, e0, s, 0 + 13) 0 + (13) 0

maior que 100% é menor que 100% (f8, e0, s, 0 + 13) 0 + (13) 0

maior que 100% é menor que 100% (f8, e0, s, 0 + 13) 0 + (13) 0

maior que 100% é menor que 100% (f8, e0, s, 0 + 13) 0 + (13) 0

maior que 100% é menor que 100% (f8, e0, s, 0 + 13) 0 + (13) 0

maior que 100% é menor que 100% (f8, e0, s, 0 + 13) 0 + (13) 0

maior que 100% é menor que 100% (f8, e0, s, 0 + 13) 0 + (13) 0

maior que 100% é menor que 100% (f8, e0, s, 0 + 13) 0 + (13) 0

maior que 100% é menor que 100% (f8, e0, s, 0 + 13) 0 + (13) 0

maior que 100% é menor que 100% (f8, e0, s, 0 + 13) 0 + (13) 0

maior que 100% é menor que 100% (f8, e0, s, 0 + 13) 0 + (13) 0