K25 - Teoria gier

Friday, 29 April 2022 17:10

. . . . 5 3 6 1 4 . . .

4 5 3 6 1 4

l r

The lines at the air pump at your gas station are getting too long! You want to optimize the process to help customers more quickly inflate their tires, sports balls, giant parade balloon animals, and other products.

The pump is automatic: you set the pressure to a specific number of pascals and plug the pump into the inflatable product, and it will inflate as needed to that exact pressure. There are only two buttons on the pump: up and down. They increase and decrease the target pressure, respectively, by exactly 1 pascal.

There is a line of $\mathbf{N}$ customers, each of whom brings exactly $\mathbf{P}$ products that they need to get inflated by the pump. You know the target pressure of each product. You can inflate the products from a customer in any order you want, but you cannot change the order of the customers. Specifically, you must inflate all products from the i-th customer before inflating any from the $(i+1)$-th customer. In between handling two products, if those two products have different target pressures, you need to use the buttons on the pump.

The pump is initially set to 0 pascals, and it can be left at any number after all products of all customers have been inflated. If you order the products of each customer optimally, what is the minimum number of button presses you need?

$\longrightarrow$ $\downarrow$

30 10 40 → 10 40

20 50 60 → 20 60

60 60 50 → 50 60

110

$O(n \cdot p)$ $n < 10^6$ $p < 100$

$x \to \min \to \max$

$x \to \max \to \min$

dp[n][2]

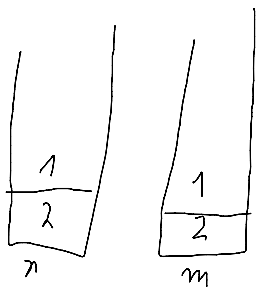
nr klienta — koniec w min/max

x min max

dp[n][0] = min(dp[n−1][0] + dist(min P[n−1], max P[n]) + dist(max P[n], min P[n]),
min   dp[n−1][1] + dist(max P[n−1], max P[n]) + dist(max P[n], min P[n]))

dist(a, b)
  abs(b − a)

dp[n][1] = min(. . . + ... , min P[n] . . . .



A: 2
B: 4

n · m
rec(0, 0) = 0
rec(a, b)

dp[n][m] − przewaga/strata gracza
dp[0][0] = 0
dp[a][b] = max(val[a] − dp[a−1][b],
    val[b] − dp[a][b−1]

$rec(0,0) = 0$

$rec(a, b)$

$- rec(a-1, b) + val[a]$

$- rec(a, b-1) + val[b]$

$\Rightarrow$

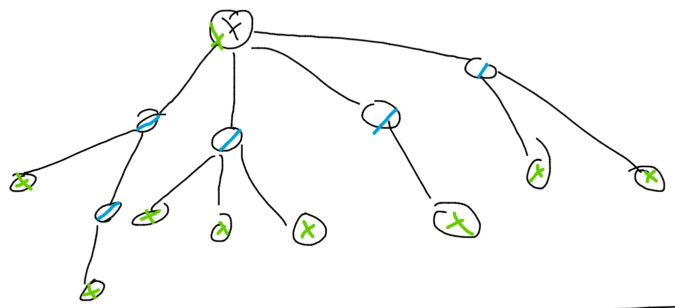
$dp[0][0] = 0$

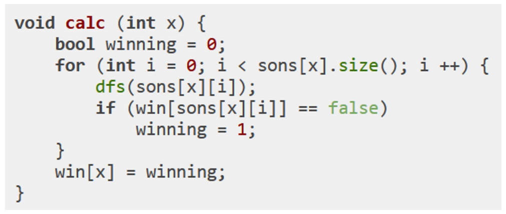
$dp[a][b] = \max\left(\begin{matrix} val[a] - dp[a-1][b], \\ val[b] - dp[a][b-1] \end{matrix} \right)$

$x + x - dp[n][m] = sum.$

$2x = sum + dp[n][m]$

$x = \frac{sum + dp[n][m]}{2}$





```
void calc (int x) {
    bool winning = 0;
    for (int i = 0; i < sons[x].size(); i ++) {
        dfs(sons[x][i]);
        if (win[sons[x][i]] == false)
            winning = 1;
    }
    win[x] = winning;
}
```

Kule n, m

-1 biała (-2 białe +1, -czarna -biała +czarna)

-2 czarne +biała

$m = 2k + 1$ ✗

$m = 2k$ ✓

Dzielniki

$\cancel{1}\ \cancel{2}\ 3\cancel{4} \ldots\ 100 \rightarrow$ ✗

⋮

$2\ 3 \ldots\ 100$ → wygrana, jeśli wyjmiemy ✗

→ przegrana

$1 \rightarrow 1$

$12 \rightarrow 2$

$123 \rightarrow 1 \rightarrow 2/3 \rightarrow 3/2$

$1234 \rightarrow 2 \rightarrow 3/4 \rightarrow 4/3$

pierwszy zawsze wygra