

Teleport – Solutie

Solutia se bazeaza pe programare dinamica. Construim matricea de dinamica $D[i][k]$ = Numarul minim de pasi pentru a ajunge in celula i folosind k teleportari personale.

$$D[i][k] = \begin{cases} \min(D[i-1][k], D[j][k]) + 1, \text{ unde } j + v[j] = i. \text{ Pentru } k = 0. \\ \min(D[i-1][k], D[j][k], D[i-K][k-1]) + 1, \text{ unde } j + v[j] = i \text{ si } i - K \geq 0. \text{ Pentru } k > 0. \end{cases}$$

Solutia este $\min(D[N][i])$, unde $i=0..K$

Solutie optima. Complexitate $O(N)$ – 100 Puncte

In timp ce se citesc valorile pentru celule, se actualizeaza costurile celulelor in care se poate ajunge din cea curenta:

Pseudocod:

```
for i = 1..N {
    citeste(x);

    for k = 0..K {
        // Pas
        if (i + 1 <= N + 1 && D[i + 1][k] > D[i][k] + 1)
            D[i+1][k] = D[i][k] + 1;
        // Teleportare podea
        if (i + x <= N + 1 && D[i + x][k] > D[i][k] + 1)
            D[i + x][k] = D[i][k] + 1;
        // Teleportare personala
        if (i + K <= N + 1 && k < K && D[i + K][k + 1] > D[i][k] + 1)
            D[i + K][k + 1] = D[i][k] + 1;
    }
}
```

Solutie optima. Complexitate $O(N)$, cu constanta mare – 100 Puncte

Ne folosim de restrictia ca valorile scrise pe podea sunt maxim 100. Pentru a calcula $D[i][k]$ iteram in spate maxim 100 de celule. De exemplu pentru a considera teleportarea de pe podea: $D[i][k] = \min(D[j][k]) + 1$, unde $j = i-101 .. i-1$ si $j + v[j] = i$.

Solutie complexitate $O(N^2)$ – 60 Puncte

Pentru a calcula $D[i][k]$ iteram in spate pana la prima celula.

Solutie backtracking – 30 Puncte