Solutie - problema **cmin**

Autor: **prof. *Constantin Gălăţan***

C. N. "Liviu Rebreanu" Bistriţa

1. O soluţie bazată pe simulare, prin care, pe fiecare coloană se plasează jetoanele în toate modurile posibile, cu ajutorul unui algoritm de backtracking, va primi între **20** si **30** de puncte.

2. O soluţie bazată pe recursie şi memoizare, care calculează tabloul `c`, avand semnificatia: `c[j][k]` = costul minim necesar pentru a plasa jetoane pe coloanele `[j..n-1]` şi primele `n/2` linii, dacă se plasează `k` jetoane pe coloana `j`, va obţine maximum **50** de puncte, în funcţie de optimizări.

3. O soluţie de complexitate `O(n*n*N)` cu programare dinamică va obţine **100** de puncte.
Ideea este de a calcula tabloul `c`, definit astfel: `c[j][k]` = costul minim necesar pentru a plasa `k` jetoane pe randurile `1..n / 2` şi coloanele `1..j` (inclusiv `j`). Se precalculează de asemenea tabloul `cc`, cu semnificaţia `cc[j][k]` – costul minim ca pe coloana `j` să avem `k` jetoane plasate pe liniile `1.. n/2`. Relaţia de recurenţă este:

```
c[j][p+k] = min(c[j][p+k], c[j-1][p] + cc[j][k])
```

unde:

`j` este coloana curentă (`1 ≤ j ≤ n`),

k este numarul de jetoane plasate pe coloana j (`0≤ k ≤ n / 2`),

p este numarul de jetoane plasate pe coloanele 1, .., j -1

si liniile `1,.., n / 2`. (`0 ≤ p ≤ N / 2 + 1`)

`N` este numărul total de jetoane de pe tabla de joc.

4. Soluţia descrisă mai jos, propusă de prof. ***Piţ-Rada Ionel Vasile*** este bazată pe o strategie greedy, şi obţine **100** de puncte.

Numărăm valorile `1` din cele două "jumataţi" ale matricei şi obţinem `nr1` şi `nr2`. Putem presupune ca `nr1>nr2` celelalte cazuri rezolvându-se analog..

Se observă uşor că vor trebui mutate, în ultimele `n/2` linii ale matricei,

`(nr1-nr2)/2` valori egale cu 1 dintre cele nr1 valori 1 aflate in primele n/2 linii ale matricei .Mutările vor fi efectuate in ordine crescatoare a costurilor.

(1) *Dacă se doreşte efectuarea unei mutări intr-o coloana c şi se pot efectua mutări în acea coloană, atunci vom alege mutarea de cost minim din coloana c care se obtine prin schimbarea dintre a[i][c]==1 si a[j][c]==0 unde i este maxim si i<=n/2 respectiv j este minim si j>=1+n/2.*

Deoarece mutarile vor fi efectuate în orice coloană conform cu (1), atunci în fiecare coloană mutările care vor fi efectuate nu vor avea “suprapuneri”. Putem parcurge astfel pentru fiecare coloană toate mutările posibile şi să contorizăm costurile la nivelul matricei. Se va obţine astfel un vector `nrcost[i]`=număr mutari de cost `i` din care se va obtine usor costul optim.

Complexitatea algoritmului este $O(n^2)$