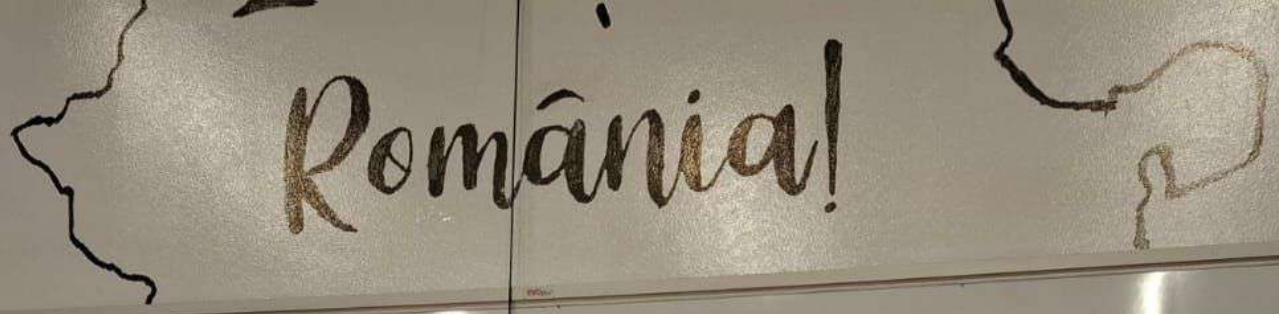
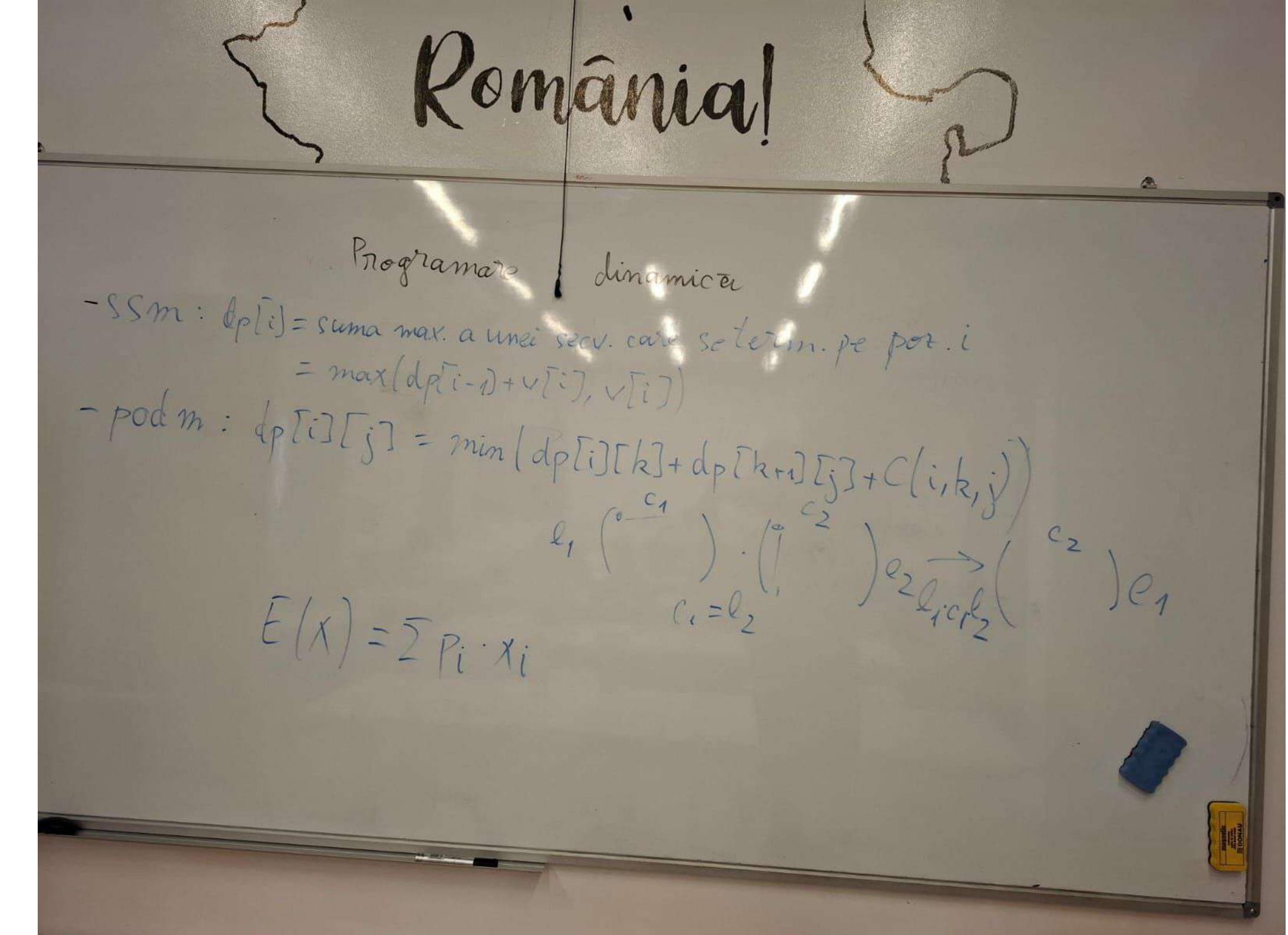
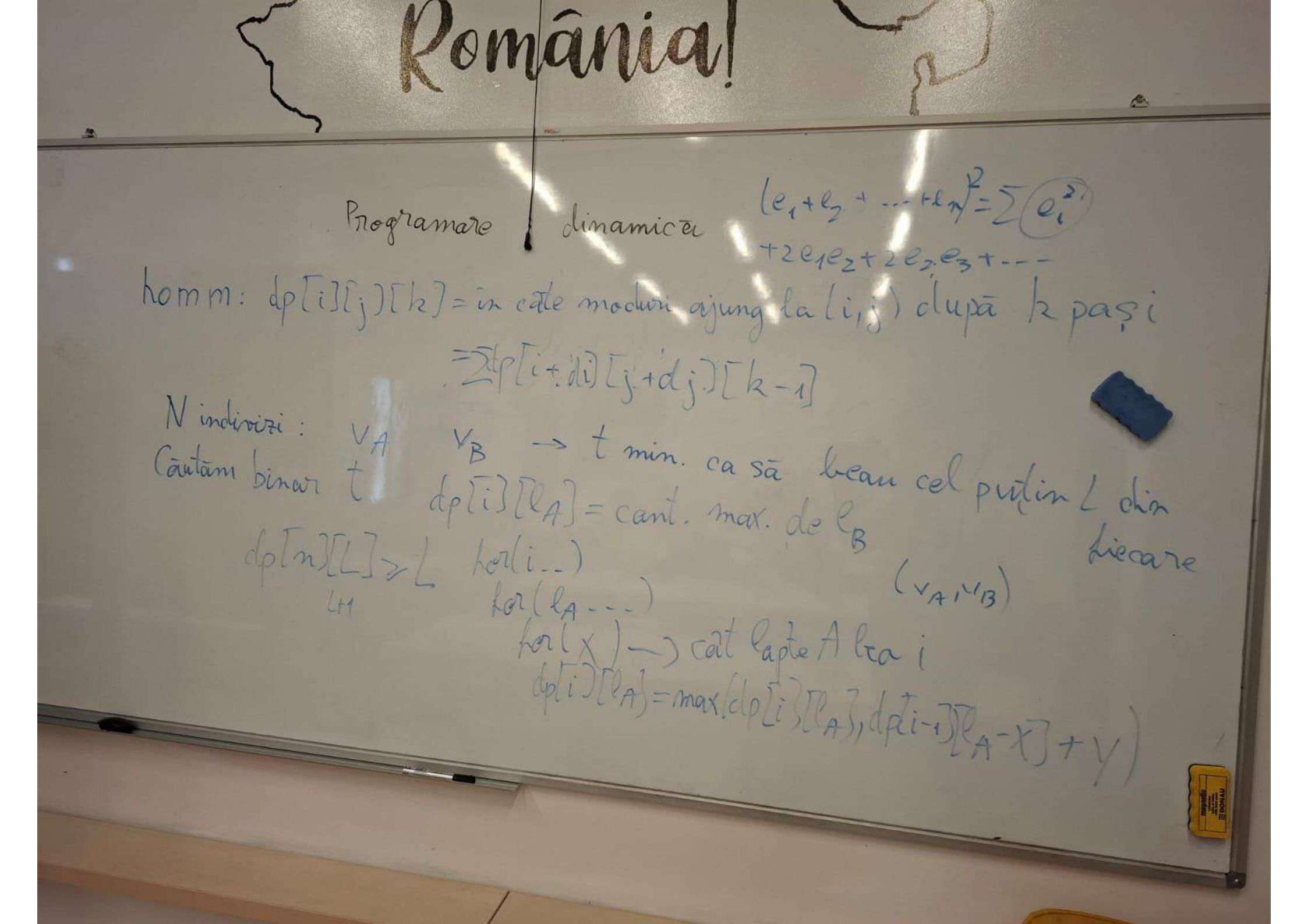


Programare dinamică

- Rucsac: $dp[i][j] = \max(dp[i-1][j], dp[i-1][j-w[i]] + p[i])$
$\{(w_i, p_i)\}$, G,
- SCC m -> dp[i] = cel mai lung subșir care se term. pe poz i
best[lg] = cea mai mică val. la finalul unui subșir de lungime lg
- Ciclu hamiltonian de cost min -> pe măști, grafuri
- cel mai lung subșir comun -> $dp[i][j] = \max(dp[i-1][j], dp[i][j-1], dp[i-1][j-1]+1)$ $a[i] = b[j]$
A, B


România!
Programare dinamică
- SSM: dp[i] = suma max. a unei secv. care se termin. pe poz. i
= max(dp[i-1] + v[i], v[i])
- pod m: dp[i][j] = min(dp[i][k] + dp[k+1][j] + C(i,k,j))
E(X) = Σ Pi · Xi


România!
Programare dinamică
$(e_1+e_2+\dots+e_n)^2=\sum e_i^2$
$+2e_1e_2+2e_2e_3+\dots$
homm: $dp[i][j][k]$ = în câte moduri ajung la (i,j) după k pași
$=\sum dp[i+di][j+dj][k-1]$
N indivizi: V_A V_B → t min. ca să fie cel puțin L din fiecare
Căutăm binar t
$dp[i][l_A]$ = cant. max. de l_B
$dp[n][L] \geq L$
$L+1$
(V_A, V_B)
for(i...)
for(l_A...)
for(x) → cât lapte A îra i
$dp[i][l_A]=\max(dp[i][l_A], dp[i-1][l_A-x]+y)$

Romania!

Programare dinamică

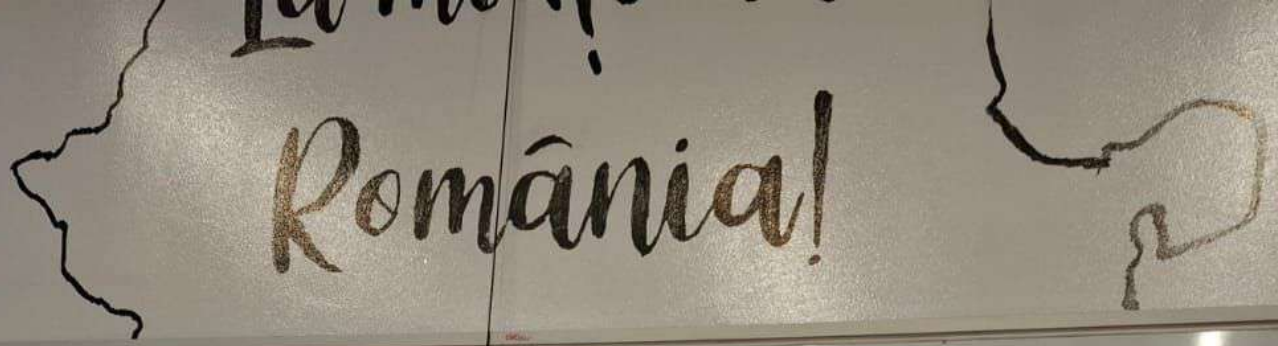
last[i] = ultimul cub înainte de i

cuburi 5 : dp[i] = lungimea maximă a unui subșir a.î. cubul i este ultimul

pair { dp2[i] = ——— a.î. ultimul cub are va
prv[j] = care e ultimul cub pt. dp2[j]
for(i)

for(val : v[x])
if(dp2[val] + 1 > dp[i]) { dp[i] = dp2[val] + 1;
last[i] = prv[val]

for(val : v[x])
dp2[val] = dp[i]
prv[val] = i



Programare dinamică

Int/m → pb spectacolelor cu pauză la mijloc

- sortăm după l_g.

$dp[i]$ = nr. maxim de intervale de la mijloc

$dp2[j]$ = nr. maxim până la spectacolul j

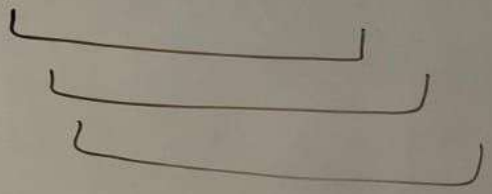
$= \max(dp2[k] + dp[j])$

k și j nu se suprapun

România!

Programare dinamică

trans: $sp[i][0/1]$



$dp[i]$ = costul min să transport primele i

$= \min(dp[j] + \min(sum(j+1 \dots i)(0/1)))$

$= \min(dp[j] + \min(sp[i][0] - sp[j][0], sp[i][1] - sp[j][0])) + T$

$= \min\begin{pmatrix} sp[i][0] + \underbrace{dp[j] - sp[j][0]}_{} \\ sp[i][1] + \underbrace{dp[j] - sp[j][1]}_{} \end{pmatrix} + T$