DESCRIERE SOLUŢII

PROBLEMA 1 LIFT

Autor: prof. Iulia Lincan
Colegiul Național ”Mihai Viteazul”, Ploiești

Notăm cu *ans[i]* numărul de posibilităţi în care pot urca cu liftul *i* persoane care au ajus în hol în acelaşi timp. Formula de recurenţă este:

$$ans[0] = 1$$
$$ans[i] = \sum_{k=1}^{i} ans[i-k] * C_i^k$$ unde C_i^k reprezintă combinări de i luate câte k, pentru orice $i \geq 1$

Fiind N teste, se precalculează vectorul *ans* pentru orice valoare a lui *i* de la 1 la 10000. Se utilizează un *map* pentru a reţine în timpul citirii din fişier momentelor de timp şi frecvenţele lor pentru cele N persoane. Rezultatul problemei este dat de produsul valorilor din vectorul *ans* pentru aceste frecvenţe.
Algoritmul are complexitatea $O(N^2)$.

PROBLEMA 2 CRIPTO

Autori:
elevii Cristian Dinu şi Matei Banu
Colegiul Național ”Mihai Viteazul”, Ploiești

Se observă în primul rând că spațiile se tranforma în trecere la rând nou.
Apoi, e important de remarcat că se folosește codul ASCII standard, și se observă că este o diferență constantă dintre literele din mesajul intial și mesajul criptat. Această este fie negativă, fie pozitivă, în funcție de poziția pe care se află litera : pară sau impara. De asemenea, spațiile nu se adaugă la poziție.
Pentru a determină diferența, se iau în calcul prima și ultima literă ale mesajului, și se adună numerele lor de ordine în alfabet. (se consideră că A este 1).

PROBLEMA 3 FTC

Autor: prof. Alice Georgescu
Colegiul Național ”Mihai Viteazul”, Ploiești

Soluția problemei se bazează pe următoarea observație:
Obs: Datorită faptului că două zone sunt vecine doar dacă au cel puţin o latură comună, numărul minim de zone pe care le traversează este dat de numărul de drepte care despart cele 2 puncte. La acestea se adaugă 1 corespunzător zonei de plecare.

PROBLEMA 4 BACTERIE

Autor:elev Matei Banu
Colegiul Național ”Mihai Viteazul”, Ploiești

Soluție 30 de puncte
Generăm la fiecare pas a, b și c de la 1 la n.

Soluție 100 puncte
Problema se rezolvă folosind exponențierea rapidă de matrici.
Construim o matrice cu a, b și c pe prima linie. Pentru fiecare tip de pas, există o matrice pe care o înmulțim la prima pentru a obține noile a, b și c. Notăm cu A, B și C cele trei matrici.
Matricea inițială trebuie înmulțită cu (A*B*C)^(n/3) și apoi cu A, dacă n%3=1 sau A*B dacă n%3=2.

PROBLEMA 5 MASINA

Autor: prof. Daniela Lica
Colegiul Național ”I.L.Caragiale”, Ploiești

Soluția problemei se bazează pe următoarele două observații:
Obs1: Există soluție optimă în care cel puțin un bec se va luă imediat ce se aprinde. Prin urmare, putem fixă acest bec.
Obs2: Având acest bec fixat, pentru fiecare dintre celelalte becuri putem determină un interval de valori pe care viteză trebuie să le ia că să vedem becul curent aprins.

PROBLEMA 4 SPIRLA

Autor:elev Cristian Dinu
Colegiul Național ”Mihai Viteazul”, Ploiești

În primul rând, se observă că pentru fiecare aparat i (1<=i<=n) trebuiesc calculate combinări de Ai luate câte Bi. Acest număr se înmulțește cu $(Xi)^p$. Acum, dintre acestea, se cere să se aleagă maxim k, optim, astfel încât produsul lor modulo M să fie maxim.

Soluție 10 puncte
Se va folosi metoda backtracking și se vor genera toate posibilitățile de a alege maxim k combinări. Combinările vor fi destul de mici încât să fie calculate trivial, după formulă.

Soluție 100 de puncte
Pentru a calcula optim numerele implicate, se va folosi invers modular pentru combinări, reținându-se un vector în care f[i] = i!, pentru a putea calcula orice combinare în complexitate log. Pentru Xi^p, cât și pentru invers modular(a ridicat la m-2, pentru inversul modular al lui a funcție de m) se va folosi ridicarea la putere în timp logaritmic.
Pentru a combina optim numerele obținute, se va folosi raționamentul de la problema rucsacului.
d[i], i – modulul curent, d[i] – numărul minim de numere înmulțite pentru a formă modulul respectiv.
Pentru că în momentul în care la d[i] adăugăm un alt număr, modulul poate fi atât mai mic, cât și mai mare, trebuie ținuți 2 vectori, ambii identici inițial, d1, d2, inițializați cu un număr mai mare decât n. Se parcurge d1, făcându-se update din d1[i] în d2[i*număr_curent%M], dacă d1[i]<k.

După fiecare parcurgere a d1, d2 se copiază în d1, păstrându-i identici pentru fiecare pas. Răspunsul se va află în d1[i], cu i maxim, cu proprietatea că d1[i] <=k.
Observația care face soluția de mai sus să treaca de la 65 de puncte la 100 este că dacă în vectorul de factoriale f, poziția depășește valoarea m, atunci toate numerele vor fi 0(de ex: combinări de 14 luate câte 14 mod 11 va da 0). Aceasta situatie este evitata daca se calculează vectorul, f[i]=f[i-1] * q (in loc de f[i-1]*i), unde q = i / cmmdc(i,m).