OLIMPIADA NAȚIONALĂ DE INFORMATICĂ 2026, ETAPA JUDEȚEANĂ
CLASELE 11-12
DESCRIEREA SOLUȚIILOR

COMISIA ȘTIINȚIFICĂ

Problema 1: KNIGHT

Propusă de: Lucian-Andrei Badea, TU Delft

În aceasta problemă avem un arbore de N noduri înrădăcinat în nodul 1 reprezentat prin vectorul de părinți $p_2, \ldots, p_N$ și avem de rezolvat 3 cerințe.

Cerința 1. La cerința 1 trebuie să numărăm câte noduri au exact un fiu. Pentru a putea calcula acest lucru, putem calcula $nrfii[i]$ = numărul de fii ai nodului i și apoi vom număra câte noduri au $nrfii[i] = 1$.

Cerința 2. La cerința 2 avem Q perechi de noduri (a, b) și pentru fiecare trebuie să aflăm dacă putem ajunge de la nodul a la nodul b făcând mutări de cal. Pentru a putea face acest lucru, ne vom baza pe următoarele observații:

(1) Dacă facem o mutare de cal de la nodul x la nodul y, atunci adâncimea nodului y este cu 1 mai mică decât adâncimea nodului x. De asemenea, nodul y este sigur diferit de nodul 1, rădăcina arborelui.
(2) Dacă facem o mutare de cal de la nodul x la nodul y, atunci trebuie ca p_y să aibă cel puțin 2 fii, pentru că altfel nu am putea coborî în nodul y.
(3) Astfel, trebuie ca adâncimea nodului b să fie **strict mai mică** decât adâncimea nodului a.
(4) Părintele nodului b trebuie să se afle pe lanțul de la nodul a la nodul 1 (cu alte cuvinte, p_b este strămoșul comun al nodurilor a și b).
(5) Nodul b trebuie să **nu** fie strămoș al nodului a.
(6) Având în vedere că toate condițiile de mai sus trebuie respectate, trebuie ca niciun nod de pe lanțul de la p_{p_a} la p_b să nu aibă exact un fiu.

Mai mult, se poate demonstra că se poate ajunge de la nodul a la nodul b făcând mutări de cal dacă și numai dacă condițiile $(3) - (6)$ țin. Astfel, putem face o parcurgere DFS în care pentru fiecare nod x notăm timpul de intrare $tin[x]$ ca fiind momentul când ajungem prima oara la nodul x și timpul de ieșire $tout[x]$ ca fiind momentul când ieșim din nodul x. Astfel, pentru a vedea dacă x este strămoș al lui y, trebuie sa vedem dacă $tin[x] \leq tin[y]$ și $tout[y] \leq tout[x]$ (adică dacă am intrat în y după x și dacă am ieșit din y înaintea lui x). În plus, putem calcula și adâncimea fiecărui nod pentru a testa a treia observație. Pentru a vedea că ultima observație este respectată putem să ne menținem $nr1[x]$ ca fiind numărul de noduri cu exact 1 fiu pe lanțul de la x la rădăcină. Dacă $nr1[p_{p_a}] = nr1[p_b]$, atunci toate nodurile de pe lanțul de la p_{p_a} la p_b au cel puțin 2 fii.
Dacă toate aceste verificări sunt adevărate, atunci răspunsul pentru perechea (a, b) este 1, altfel 0. Complexitatea este de $O(N)$ pentru precalculările înălțimilor nodurilor și a $tin[x]$, $tout[x]$ și $nr1[x]$ și $O(1)$ pentru fiecare dintre cele Q perechi, deci $O(N + Q)$ în total.

Cerința 3. La cerința 3 avem Q perechi de noduri (a, b) și pentru fiecare trebuie să calculăm în câte moduri putem ajunge de la a la b făcând mutări de cal.
În primul rând, trebuie să facem toate verificările de la cerința 2 pentru a vedea dacă putem ajunge de la a la b. Apoi, fiindcă după fiecare mutare de cal adâncimea nodului curent scade

cu 1, conform observației (1), noi ne vom plimba prin fiecare nod de pe lanțul de la a la p_b. Fie v nodul care este fiu al nodului p_b și se află pe lanțul de la a la p_b. Atunci, pentru fiecare nod x de la p_{p_a} la v vom avea $nrfii[x] - 1$ alegeri distincte de noduri în care să coborâm. Astfel, răspunsul, în cazul în care putem ajunge de la a la b este produsul pe lanț $\prod_{x=p_{p_a}}^{v}(nrfii[x]-1)$. Pentru a putea calcula acest produs eficient, vom precalcula $prod[i] = \prod_{x=i}^{1}(nrfii[x]-1)$, adică produsul de $nrfii[x] - 1$ de pe lanțul de la i la rădăcină. Trebuie să avem grijă la nodurile cu exact un fiu, întrucât acestea contribuie cu 0 la produs. Astfel, atunci când ajungem în ele în parcurgerea DFS, nu le vom lua în calcul la produs. Astfel, $prod[i]$ va menține de fapt doar produsul pentru nodurile de la i la rădăcină care au cel puțin 2 fii. Așadar, răspunsul final pentru o pereche (a, b) va fi $\frac{prod[p_{p_a}]}{prod[p_b]}$ modulo 10^9+7, care se poate calcula folosind inversul modular al lui $prod[p_b]$. Astfel, complexitatea este de $O(N)$ pentru precalculări și $O(Q \log MOD)$ datorită calculului inversului modular pentru fiecare dintre cele Q perechi, deci $O(N+Q \log MOD)$ în total.

Bonus. Cerința 3 se poate rezolva și în complexitate liniară. Aceasta va rămâne ca exercițiu pentru cititor.

Problema 2: ROTAȚII

Propusă de: Verzotti Matteo-Alexandru, Universitatea București

În continuare definim un *pas* ca fiind o pereche formată dintr-o translație și o rotație realizate succesiv.

Cerința 1. Pentru această cerință vom lua alternativ câte o translație și o rotație și vom simula operațiile folosind tabelul din enunț.

Cerința 2. Pentru această cerință este nevoie să facem întâi câteva observații. Reamintim distanța Manhattan între două puncte (x_0, y_0) și (x_1, y_1) ca fiind $|x_0 - x_1| + |y_0 - y_1|$. Notăm cu $d = |a| + |b|$ distanța Manhattan între $(0, 0)$ (crâșmă) și (a, b) (casa lui Gigel).

Observație 1. *Un pas ne schimbă întotdeauna distanța Manhattan cu $+1$ sau -1 (exceptând cazul când suntem în origine).*

Demonstrație. Se poate observa că distanța Manhattan este invariantă la rotație. Doar translația o afectează, iar dintr-un punct (x, y) ne putem muta doar în $(x-1, y), (x+1, y), (x, y-1)$ și $(x, y+1)$. Dintre acestea, minim unul este mai aproape de origine, excepție făcând cazul când suntem deja în origine, iar toate schimbă distanța cu Manhattan cu exact 1. □

Observație 2. *Paritatea distanței Manhattan până la origine după efectuarea a N pași este egală cu paritatea lui N.*

Demonstrație. Reiese din observația anterioară. □

Prin urmare, dacă $d > N$ sau N și d au parități diferite, atunci nu avem soluție. În continuare, vom demonstra că negația acestor condiții este suficientă pentru existența soluției.

Observație 3. *Oricare ar fi șirul de translații, există un șir de rotații astfel încât după N pași să ajungem în orice punct din plan ce are distanța Manhattan $d \in \{N, N-2, \ldots, N \mod 2\}$.*

Demonstrație. Vom demonstra în mod inductiv că după N pași putem ajunge în orice punct din plan ce are distanța Manhattan N față de origine. Dacă avem la dispoziție mai mulți pași decât necesar, adică $d < N$, ne putem învârti în jurul lui $(0, 0)$ cu un număr par de pași.

Caz de bază: Folosind 0 pași putem ajunge în $(0, 0)$ ce are distanță 0 față de origine (evident).

Pas inductiv: Presupunem că folosind $N-1$ pași putem ajunge în toate punctele de distanță Manhattan $N-1$ și demonstrăm că folosind N pași putem ajunge în toate punctele de distanță Manhattan N. Fie (a, b) un punct de distanță Manhattan N. Presupunem fără a restrânge

generalitatea că $a, b > 0$, deci $a + b = N$ (celelalte cazuri se demonstrează analog). Fie tr_N translația de la pasul N. Avem 4 cazuri:

(1) $tr_N =$ `N`. Atunci din $(a, b-1)$ aplicând pasul $(\texttt{N}, 0)$ ajungem în (a, b), iar $(a, b-1)$ are distanța Manhattan $a + b - 1 = N - 1$.
(2) $tr_N =$ `S`. Atunci din $(-a, -b+1)$ aplicând pasul $(\texttt{S}, 180)$, ajungem în (a, b), iar $(-a, -b+1)$ are distanța Manhattan $|-a| + |-b+1| = a + b - 1 = N - 1$.
(3) $tr_N =$ `E`. Atunci din $(a-1, b)$ aplicând pasul $(\texttt{E}, 0)$, ajungem în (a, b), iar $(a-1, b)$ are distanța Manhattan $a - 1 + b = N - 1$.
(4) $tr_N =$ `V`. Atunci din $(-a+1, -b)$ aplicând pasul $(\texttt{V}, 180)$, ajungem în (a, b), iar $(-a+1, -b)$ are distanța Manhattan $|-a+1| + |-b| = a - 1 + b = N - 1$.

Astfel, am demonstrat că putem ajunge în orice punct de distanță Manhattan N. □

În concluzie, putem ajunge în (a, b) din N pași dacă și numai dacă sunt satisfăcute simultan următoarele două condiții, indiferent de șirul de translații dat:

(1) $d \leq N$
(2) $d \equiv N \pmod 2$

Cerința 3. Pentru ultima cerință, este suficient să pornim din punctul (a, b) și să aplicăm rotațiile conform demonstrației **Observației 3** de mai sus, mergând de fiecare dată într-un punct cu distanță Manhattan mai mică. Trebuie avută grijă în cazul în care avem la dispoziție mai mulți pași decât este necesar. În acest caz, putem folosi la final câte doi pași pentru a ne muta din origine într-o poziție vecină și apoi pentru a reveni în origine. Repetăm acest procedeu de câte ori este nevoie. De menționat că nu este neapărat necesar să analizăm cazuri; putem verifica la fiecare pas toate cele patru rotații posibile și să o alegem pe cea care duce la o distanță mai mică.

Complexitatea soluției este liniară pentru toate cerințele.

Problema 3: ELIBERARE

Propusă de: Szabó Zoltan — Inspectoratul Școlar Județean Mureș, Târgu Mureș

Pentru simplitate în scrierea complexităților timp, vom presupune că $N = M$.

Subtask 1: Toate mașinile sunt orientate în aceeași direcție (40 de puncte). Folosind două cicluri for, unul pentru linii și unul pentru coloane, alegând parcurgerea corectă cu pasul $+1$ sau -1, se pot elibera toate mașinile. Complexitate timp $O(N^2)$.

Subtask 2: Toate mașinile sunt orientate în doar 2 dintre cele 4 direcții posibile (30 de puncte).
Avem șase posibilități distincte, care se încadrează în două subcategorii:

(1) Cele două direcții sunt perpendiculare: Nord-Est, Nord-Vest, Sud-Est, Sud-Vest. În aceste cazuri, la fel ca la subtaskul 1, cu două cicluri imbricate se poate genera o soluție validă. O altă posibilitate de rezolvare se bazează pe parcurgerea matricei paralel cu diagonala principală sau secundară.
(2) Cele două direcții sunt opuse: Nord-Sud, Est-Vest. În acest caz, pentru fiecare linie (Est-Vest) sau coloană (Nord-Sud) vom executa parcurgeri dinspre cele două extremități către interiorul liniei/coloanei, până la întâlnirea celor două mașini cu direcția opusă.

Complexitate timp: $O(N^2)$.
Observație: Rezolvarea de la subtaskul 1 poate să prindă câteva cazuri și de la subtaskul 2.

Subtask 3: $N, M \leq 100$ **(12 puncte).**
Soluția brută. În mod repetat se parcurg toate elementele matricei. Pentru fiecare masină se verifică dacă poate ieși din parcare. Algoritmul se reia în mod repetat până când parcarea se golește complet. Complexitate timp este $O(N^5)$ sau $O(N^4)$ sau $O(N^3)$, în funcție de implementare.

Subtask 4: Fără restricții suplimentare (18 puncte).
Parcarea se poate interpreta ca un graf orientat în care nodurile sunt mașinile, iar fiecare mașină este conectată prin arce la mașinile care o blochează din a fi eliberată (de exemplu, pentru o mașină de tip >, acestea sunt mașinile aflate la dreapta ei, pe aceeași linie). Dacă graful ar conține un circuit, atunci problema nu ar avea soluție. Conform enunțului, acest caz este exclus. Pentru a obține o soluție optimă, vom parcurge elementele matricei. Dacă găsim o mașină nerezolvată, printr-o parcurgere în adâncime descoperim toate mașinile care trebuie eliberate înaintea celei de pe poziția curentă. Lanțul de mașini este memorat într-o stivă. Deoarece graful nu conține cicluri, lanțul generat va avea întotdeauna lungime finită, iar ultima mașină introdusă în stivă va fi validă pentru eliberare. O astfel de soluție poate avea complexitatea $O(N^3)$ sau $O(N^2)$. Pentru a obține complexitatea $O(N^2)$, vom evita reținerea întregului graf și vom memora pentru fiecare mașină în parte doar prima mașină care o blochează, sub forma unor legături dublu înlănțuite pe direcțiile Nord–Sud și Est–Vest. La ștergerea unei mașini eliberate din matrice, refacem legăturile dintre mașinile rămase în timp $O(1)$.

La acest subtask sunt incluse teste cu $N, M \leq 800$, respectiv cu $1000 \leq N, M \leq 1100$, pentru a diferenția soluțiile în $O(N^3)$ de cele în $O(N^2)$.

Echipa

Problemele pentru această etapă au fost pregătite de:

- Szabó Zoltan, Inspectoratul Școlar Județean, Târgu-Mureș
- Panaete Adrian, Colegiul Național "A.T. Laurian", Botoșani
- Constantinescu Andrei-Costin, ETH, Zürich, Elveția
- Badea Lucian-Andrei, TU Delft, Olanda
- Verzotti Matteo-Alexandru, Universitatea București
- Ariciu Toma, Universitatea Națională de Știință și Tehnologie Politehnica, București
- Bogdan Vlad-Mihai, Universitatea București
- Spătărel Dan-Constantin, București
- Ciucu Mihai, CS Academy, București
- Posdarascu Eugenie-Daniel, Youni, București
- Todoran Alexandru-Raul, Harvard University, Cambridge, Massachusetts
- Stănescu Matei-Octavian, Universitatea Națională de Știință și Tehnologie Politehnica, București
- Prosie Radu-Teodor, Universitatea București
- Ilie Luca-Mihai, École Polytechnique, Paris, Franța
- Roșu Adrian-Andrei, Universitatea Națională de Știință și Tehnologie Politehnica, București
- Haba Matei-Ionuț, Universitatea Națională de Știință și Tehnologie Politehnica, București
- Marciuc Andrei, Universitatea Națională de Știință și Tehnologie Politehnica, București
- Moca Andrei, Universitatea "Babeș-Bolyai", Cluj-Napoca
- Nuță Ștefan-Alexandru, Universitatea Națională de Știință și Tehnologie Politehnica, București
- Taschina Luca-Marian, Universitatea "Babeș-Bolyai", Cluj-Napoca
- Turcu Andrei-Cristian, Universitatea București
- Gheorghieș Petruț-Rareș, Universitatea Națională de Știință și Tehnologie Politehnica, București
- Ionescu Matei, Universitatea "Alexandru Ioan Cuza", Iași
- Ciuleanu Vlad, ETH, Zürich, Elveția