

Task 1. Monopoly

Lui **Deni** îi place foarte mult să joace monopoly. Dar durata unui joc este foarte mare – între 5 și 7 ore. Așa că **Deni** se gândește să schimbe regulile clasice. De obicei la monopoly jucătorii se deplasează dintr-o căsuță în alta, într-o anumită direcție și, după un timp, revin la poziția de start, apoi procesul se reia. În noua versiune mutarea se va face de asemenea dintr-o căsuță în alta, dar din cea curentă pot exista mai multe variante pentru următoarea mutare. **Deni** vrea să găsească legături unidirecționale între căsuțe astfel încât jucătorul să nu poată să revină într-o căsuță în care a mai fost, indiferent cum se deplasează (desigur, respectând regulile). Astfel jocul va dura mai puțin.

Ea a început deja să creeze noua tablă – a ales numărul de căsuțe **N** (căsuțele sunt numerotate de la 1 la **N**) și a făcut o listă cu **M** legături directe (fiecare legătură este unidirecțională și nu există o legătură între o căsuță și ea însăși). În cazul în care căsuța *i* este legată de căsuța *j*, atunci nu există și o legătură directă în sens invers, adică de la căsuța *j* către căsuța *i* și, de asemenea, nu pot exista mai multe legături directe de la căsuța *i* la *j*. **Deni** credea că a terminat noua tablă, dar deodată a observat că acea condiție pe care și-ar dori-o respectată (atunci când te muți dintr-o căsuță în alta folosind legăturile directe nu poți reveni într-o căsuță care a fost deja vizitată) nu e îndeplinită de lista ei de legături. Mai întâi s-a gândit să elimine o parte dintre legăturile directe, dar asta ar avea ca efect rescrierea listei, ceea ce ar putea dura foarte mult. De aceea **Deni** s-a hotărât să inverseze sensul de parcurgere pentru unele dintre legăturile directe.

Cerința

Având în vedere că joci frecvent monopoly cu **Deni**, îți dorești să o ajuți să scrie programul **monopoly**, care trebuie să îi spună ce legături să inverseze astfel încât condiția menționată să fie respectată. Programul trebuie să conțină funcția *find_reverse* care va fi compilată cu programul comisiei.

Detalii de implementare

Funcția *find_reverse* trebuie să aibă următorul prototip:

```
std::string find_reverse (int N, int M, int connections[][2]);
```

Este apelată o singură dată de programul comisiei cu 3 parametri: **N** - numărul de căsuțe de pe noua tablă, **M** - numărul legăturilor directe din lista lui **Deni** și tabloul *connections*, care are **M** linii, fiecare conținând câte două numere *x* și *y* - căsuța inițială și cea finală corespunzătoare legăturii directe. Această funcție trebuie să returneze un string binar de lungime **M** - respectând ordinea din tabloul *connections*, pentru fiecare legătură directă string-ul va conține '1' dacă legătura trebuie inversată și '0' în caz contrar. Dacă există mai multe soluții, funcția o poate returna pe oricare dintre ele.

Trebuie să încarci fișierul **monopoly.cpp**, care conține implementarea funcției *find_reverse*. Fișierul poate conține și alte funcții și secvențe de cod necesare programului, dar **trebuie să nu conțină** o funcție principală - *main*. În plus programul trebuie să nu încerce să citească sau scrie de la input-ul sau output-ul standard!

Restricții

♣ $3 \leq N \leq 5 \times 10^5$

♣ $3 \leq M \leq 1.5 \times 10^6$

Subtask-uri

Subtask	Points	N	M	Restricții suplimentare
1	0	–	–	Exemplul
2	15	≤ 7	≤ 21	–
3	40	$\leq 10^3$	$\leq 5 \times 10^3$	–
4	25	$\leq 10^5$	$\leq 5 \times 10^5$	–
5	20	$\leq 5 \times 10^5$	$\leq 1.5 \times 10^6$	–

Pentru a obține punctele corespunzătoare unui anumit subtask, soluția ta trebuie să treacă toate testele subtask-ului.

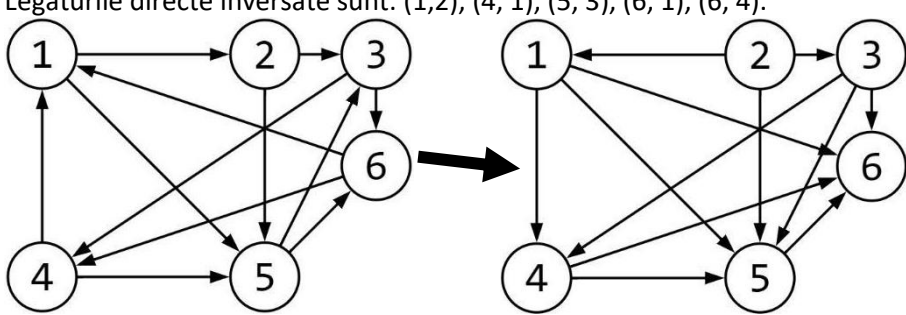
Testarea locală

Pentru testarea locală ți se pune la dispoziție fișierul **Lgrader.cpp**. Trebuie să îl copiezi în același folder ca și programul **monopoly.cpp** și trebuie să compilezi împreună fișierele **Lgrader.cpp** și **monopoly.cpp**. Astfel, vei crea un program care să verifice corectitudinea funcției tale. Programul va solicita următoarele date de la input-ul standard:

- pe prima linie: doi întregi – numărul de căsuțe N și numărul de legături directe M ale noii table.
- pe ultimele M linii: pe fiecare linie trebuie să se afle doi întregi x și y – căsuța inițială și cea finală pentru fiecare legătură directă.

Rezultatul afișat de program va fi string-ul binar găsit.

Exemplu pentru testarea locală

Intrare	Ieșire	Explicație
6 12 1 2 1 5 2 3 2 5 3 4 3 6 4 1 4 5 5 3 5 6 6 1 6 4	100000101011	<p>Legăturile directe inversate sunt: (1,2), (4, 1), (5, 3), (6, 1), (6, 4).</p>  <p>Figura de deasupra prezintă căsuțele, mai întâi cu legăturile directe din lista lui Deni și apoi, după inversarea sensului legăturilor pentru elementele '1' din string-ul rezultat. Evident, la început, folosind legăturile directe (1, 2), (2, 3), (3, 4) și (4, 1) putem porni din căsuța 1 și deplasându-ne conform acestora, vom reveni în acea căsuță. Se poate observa că în tabla de joc rezultată condiția lui Deni e îndeplinită. Se observă că există și alte soluții valide: (4, 1), (5, 3), (6, 1), (6, 4) (1, 5), (2, 3), (2, 5), (3, 4), (3, 6), (4, 5), (5, 6)</p>