Rosvo ja poliisi

Bytemoren kaupungissa rikosten määrä on saavuttamassa kaikkien aikojen huipun. Muun rötöstelyn lisäksi ryöstöjä tapahtuu joka päivä. Joka kerta kun ryöstö tapahtuu, on yksittäisen partioivan poliisin tehtävä jahdata rosvo kiinni kadunkulmia yhdistäviä kapeita kujia pitkin. Ikävä kyllä, useimmiten rosvot pääsevät pakoon takaa-ajajiltaan, koska he tuntevat kaupungin paljon poliisia paremmin.

Bytemoren kaupungin poliisilaitos järjestää kokouksen, jonka tehtävänä on vähentää rikollisuutta. Yksi aloitteista on käyttää tietokonetta apuna rosvojen jahtaamisessa. Tätä varten poliisilaitos on tehnyt tarkan kartan kaupungista. Nyt he tarvitsevat tietokoneohjelmaa jahtausstrategioiden päättämiseen.

Rosvojahti jossa yksi poliisi jahtaa yhtä rosvoa mallinnetaan seuraavasti:

1. Poliisi valitsee kadunkulman jossa partioi.
2. Rosvo valitsee kadunkulman jossa ryöstö tehdään (hän tietää aina missä poliisi on). Tästä lähtien oletetaan aina että sekä poliisi että rosvo tietävät missä kumpikin ovat.
3. Poliisi siirtyy siirrollaan joko viereiseen kadunkulmaan (sellaiseen johon kulkee nykyisestä kuja) tai odottaa paikallaan (ei siirry).
4. Rosvo siirtyy aina vuorollaan viereiseen kadunkulmaan. Huomaa, että toisin kuin poliisit, rosvot eivät voi odottaa paikallaan. Heidän vaistonsa saa heidät jatkamaan juoksua.
5. Poliisi ja rosvo tekevät siirtoja vuorotellen (aloittaen poliisista) kunnes yksi seuraavista tapahtuu:
   (a) sama tilanne toistuu (tilanne määritellään rosvon ja poliisin sijainteina ja sillä, kenen vuoro on seuraavaksi). Tämä vastaa sitä, että rosvo voi vältellä poliisia loputtomasti, joten rosvo pääsee pakoon;
   (b) poliisi ja rosvo ovat samassa kadunkulmassa kumman tahansa vuoron jälkeen. Tässä tapauksessa poliisi saa rosvon kiinni.

Tehtävä

Sinun tulee kirjoittaa ohjelma, jolle annetaan kaupungin kartta, ja joka päättelee onko poliisin mahdollista saada rosvo kiinni, ja jos on, osaa kertoa millä poliisin siirroilla rosvo saadaan kiinni.

Ohjelmasi on oletettava että rosvo liikkuu optimaalisesti.

Toteutus

Sinun tulee toteuttaa kaksi funktiota:

- `start(N, A)` joka ottaa seuraavat parametrit:

- N — kadunkulmien lukumäärä (kadunkulmat numeroidaan luvuilla $0 \ldots N-1$);
- A — kaksiulotteinen taulukko joka määrittelee kujat: Kaikille $0 \leq i, j \leq N-1$,

$$A[i,j] \text{ on } \begin{cases} \texttt{false}, & \text{jos kadunkulmia } i \text{ ja } j \text{ ei yhdistä kuja} \\ \texttt{true}, & \text{jos kadunkulmia } i \text{ ja } j \text{ yhdistää kuja} \end{cases}$$

Kaikki kujat ovat kaksisuuntaisia (siis $A[i,j] = A[j,i]$ kaikille kadunkulmille i ja j) ja mikään kuja ei liitä kadunkulmaa itseensä (siis $A[i,i]$ on `false` kaikille kadunkulmille i). Voit myös olettaa että voit saavuttaa minkä tahansa kadunkulman mistä tahansa toisesta kadunkulmasta liikkumalla kujia pitkin.

Mikäli poliisin on mahdollista saada rosvo kiinni annetussa kartassa, funktion `start` tulisi palauttaa sen kadunkulman tunnusnumero jossa poliisi päättää partioida. Muussa tapauksessa sen tulisi palauttaa -1.

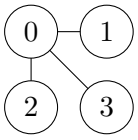
- `nextMove(R)` joka ottaa parametrina rosvon nykyisen kadunkulman tunnusnumeron R ja palauttaa sen kadunkulman tunnusnumeron missä poliisi on siirtonsa jälkeen.

Funktiota `start` kutsutaan täsmälleen kerran ennen kuin kutsuja funktioon `nextMove` tehdään. Mikäli `start` palauttaa -1, niin funktiota `nextMove` ei kutsuta. Muussa tapauksessa funktiota `nextMove` kutsutaan toistuvasti kunnes jahti on ohi. Tarkemmin ohjelma loppuu kun yksi seuraavista tapahtuu:

- `nextMove` palauttaa virheellisen siirron;
- tilanne toistuu;
- poliisi saa rosvon kiinni.

Esimerkki

Tarkastellaan oikealla olevaa esimerkkiä. Tässä tapauksessa mikä tahansa kadunkulma on hyvä aloituskohta poliisille. Jos hän aloittaa kadunkulmasta 0, hän voi odottaa ensimmäisen siirtonsa ajan ja rosvo juoksee hänen luokseen. Jos taas hän aloittaa mistä tahansa muusta kadunkulmasta, hän voi odottaa, kunnes rosvo siirtyy kadunkulmaan 0, ja siirtyä sitten sinne.



Tässä on esimerkki tapahtumien kulusta:

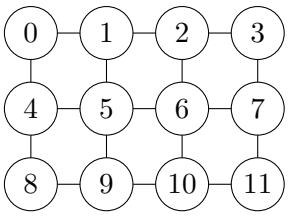
Funktiokutsu	**Palauttaa**
`start(4, [[0, 1, 1, 1], [1, 0, 0, 0], [1, 0, 0, 0], [1, 0, 0, 0]])`	3
`nextMove(1)`	3
`nextMove(0)`	0

Huomaa, että funktion `start` kutsussa ylhäällä `0` tarkoittaa samaa kuin `false` ja `1` tarkoittaa samaa kuin `true` lyhyyden vuoksi.

Pisteytys

Osatehtävä 1 (16 pistettä): $2 \leq N \leq 500$. Jokaisen kahden kadunkulman välillä on tasan yksi polku.

Osatehtävä 2 (14 pistettä): $2 \leq N \leq 500$. Kadunkulmien ja kujien verkosto muodostaa ruudukkorakenteen. Ruudukossa on vähintään kaksi riviä ja saraketta, ja kadunkulmien numerointi noudattaa alla olevaa sääntöä.



Osatehtävä 3 (30 pistettä): $2 \leq N \leq 100$.

Osatehtävä 4 (40 pistettä): $2 \leq N \leq 500$.

Ratkaisusi tulisi täyttää seuraavat kaksi vaatimusta:

1. määrittää oikein voiko poliisi saada rosvon kiinni;
2. onnistuu ottamaan rosvon kiinni tekemällä poliisin siirrot, mikäli tämä on mahdollista

Osatehtävissä 1 ja 2 ratkaisusi täytyy toteuttaa molemmat vaatimukset saadakseen pisteitä. Osatehtävissä 3 ja 4 ratkaisut jotka toteuttavat vain ensimmäisen vaatimuksen saavat 30% osatehtävän pisteistä. Mikäli ratkaisusi tähtää vain osittaisiin pisteisiin, voit lopettaa ohjelman tulostamalla virheellisen siirron (esimerkiksi palauttaa -1 funktiossa `nextMove`).

Huomaa että perusvaatimukset (aika- ja muistirajat, ei ajonaikaisia virheitä) täytyy toteutua jotta saisit yhtään pisteitä.

Rajat

Aikaraja: 1.5 s.

Muistiraja: 256 MB.

Kokeileminen

Koneellasi oleva esimerkkitarkastaja lukee tietoa standardisyötteestä. Syötteen ensimmäisellä rivillä tulee olla kokonaisluku N — kadunkulmien lukumäärä. Seuraavilla N rivillä tulee olla vierusmatriisi A. Jokaisella tällaisella rivillä tulee olla N lukua, joista jokainen on 0 tai 1. Matriisin tulee olla symmetrinen ja päädiagonaalin kaikkien arvojen tulee olla nollia.

Seuraavalla rivillä tulee olla luku 1, jos poliisi pystyy saamaan kiinni ryöstäjän, ja muuten luku 0.

Jos poliisi pystyy ottamaan rosvon kiinni, täytyy tulla vielä N riviä, jotka kuvaavat rosvon strategian. Jokaisella tällaisella rivillä tulee olla $N + 1$ kokonaislukua 0:n ja $N - 1$:n välillä. Rivin r sarakkeen c arvo, jossa $c < N$, vastaa tilannetta, jossa on rosvon vuoro, poliisi on kadunkulmassa r ja rosvo on kadunkulmassa c. Arvo kuvaa kadunkulman, johon rosvon on

siirryttävä. Päädiagonaalin arvoista ei välitetä, koska ne vastaavat tilanteita, joissa rosvo ja poliisi ovat samassa kadunkulmassa. Rivin r viimeinen arvo määrittelee rosvon aloituskadunkulman jos poliisin aloituskadunkulma on r.

Tässä on esimerkkisyöte tarkastajalle, joka kuvaa kolme kadunkulmaa, jotka on yhdistetty toisiinsa:

```
3
0 1 1
1 0 1
1 1 0
1
0 2 1 2
2 0 0 2
1 0 0 1
```

Ja tässä on syöte, joka täsmää yllä olevan tehtävänannon esimerkkiin:

```
4
0 1 1 1
1 0 0 0
1 0 0 0
1 0 0 0
1
0 0 0 0 1
2 0 0 0 2
3 0 0 0 3
1 0 0 0 1
```