Longest Trip

Organizatorii IOI 2023 au dat de belele majore! Au uitat să planifice excursia la Ópusztaszer pentru mâine. Dar poate nu e încă prea târziu ...

Sunt N atracții în Ópusztaszer indexate de la 0 la $N-1$. Unele perechi de atracții sunt conectate de **drumuri** *bidirecționale*. Fiecare pereche de atracții este conectată de cel mult un drum. Organizatorii *nu știu* care atracții sunt conectate de drumuri.

Spunem că **densitatea** rețelei de drumuri Ópusztaszer este **cel puțin** δ dacă pentru fiecare 3 atracții distincte au cel puțin δ drumuri între ele. În alte cuvinte, pentru fiecare triplet de atracții (u, v, w) pentru care $0 \leq u < v < w < N$, între perechile de atracții $(u, v), (v, w)$ și (u, w) cel puțin δ perechi sunt conectate de câte un drum.

Organizatorii *știu* un întreg pozitiv D astfel încât densitatea rețelei de drumuri este cel puțin D. Luați la cunoștință că valoarea lui D nu poate fi mai mare decât 3.

Organizatorii pot face **apeluri** la dispecerul telefonic din Ópusztaszer să descopere informații despre drumurile dintre anumite atracții. În fiecare apel, două șiruri de atractii $[A[0], \ldots, A[P-1]]$ și $[B[0], \ldots, B[R-1]]$ trebuie să fie specificate. Atracțiile trebuie să fie distincte două câte două, adică

- $A[i] \neq A[j]$ pentru fiecare i și j astfel încât $0 \leq i < j < P$;
- $B[i] \neq B[j]$ pentru fiecare i și j astfel încât $0 \leq i < j < R$;
- $A[i] \neq B[j]$ pentru fiecare i și j astfel încât $0 \leq i < P$ and $0 \leq j < R$.

Pentru fiecare apel, dispecerul raportează dacă este un drum ce conectează o atracție din A și o atracție din B. Mai exact, dispecerul iterează peste toate perechile i și j astfel încât $0 \leq i < P$ și $0 \leq j < R$. Dacă, pentru oricare dintre ele, atracțiile $A[i]$ și $B[j]$ sunt conectate de un drum, dispecerul întoarce valoarea `true`. Altfel, întoarce valoarea `false`.

Pentru fiecare apel, dispecerul raportează dacă este un drum care sa conecteze o atracție din A cu o atracție din B. Adică, dispecerul returnează `true` dacă există i și j astfel încât $0 \leq i < P$ și $0 \leq j < R$, și $A[i]$ si $B[j]$ sunt conectate de un drum. Altfel, dispecerul întoarce valoarea `false`.

Un **itinerar** de lungime l este o secvență de atracții $t[0], t[1], \ldots, t[l-1]$, unde pentru fiecare i între 0 și $l-2$ inclusiv, atracția $t[i]$ și atracția $t[i+1]$ sunt conectate de un drum. Un itinerar de lungime l se numește un **itinerar cel mai lung** dacă nu exista niciun itinerar de lungime $l+1$.

Sarcina voastră este să ajutați organizatorii să găsească un cel mai lung itinerar la Ópusztaszer făcând apeluri la dispecer.

Detalii de implementare

Ar trebui să implementați următoarea procedură:

```
int[] longest_trip(int N, int D)
```

- N: numărul de atracții la Ópusztaszer.
- D: densitatea minimă garantată a rețelei.
- Această procedură ar trebui să întoarcă un șir $t = [t[0], t[1], \ldots, t[l-1]]$, ce reprezintă un cel mai lung itinerar.
- Procedura aceasta poate fi apelată **de mai multe ori** în fiecare caz de test.

Procedura de mai sus poate face apeluri la următoarea procedură:

```
bool are_connected(int[] A, int[] B)
```

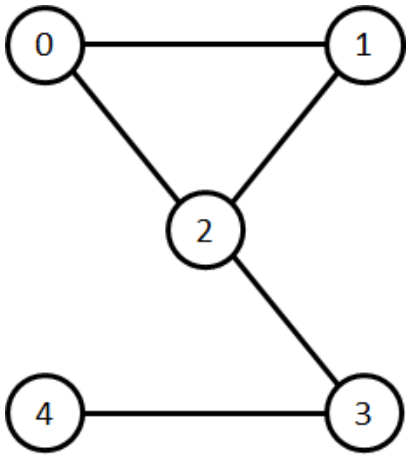
- A: un șir nevid de atracții distincte.
- B: un șir nevid de atracții distincte.
- A și B trebuie să fie disjuncte.
- Această procedură întoarce `true` dacă există o atracție din A și o atracție din B conectate de un drum. Altfel, întoarce `false`.
- Această procedură poate fi apelată de cel mult $32\,640$ ori în fiecare invocare a lui `longest_trip`, și de cel mult $150\,000$ ori în total.
- Lungimea totală a șirurilor A și B date procedurii în toate invocările nu poate întrece $1\,500\,000$.

Graderul **nu este adaptiv**. Fiecare submisie este testată pe același set de cazuri de test. Adică, valorile lui N și D, cât și perechile de atracții conectate de drumuri, sunt fixate pentru fiecare apel al lui `longest_trip` în fiecare caz de test.

Exemple

Exemplul 1

Considerați un scenariu în care $N = 5$, $D = 1$, și conecțiunile drumurile sunt precum în figura ce urmează:



Procedura `longest_trip` este apelată în felul următor:

```
longest_trip(5, 1)
```

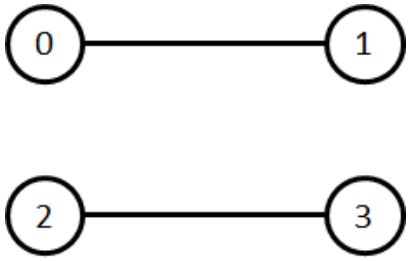
Această procedură poate face apeluri la `are_connected` în felul următor.

Apel	Perechi conectate de un drum	Valoare întoarsă
`are_connected([0], [1, 2, 4, 3])`	$(0,1)$ și $(0,2)$	`true`
`are_connected([2], [0])`	$(2,0)$	`true`
`are_connected([2], [3])`	$(2,3)$	`true`
`are_connected([1, 0], [4, 3])`	niciuna	`false`

După al patrulea apel, se întâmplă ca *niciuna* dintre perechile $(1,4)$, $(0,4)$, $(1,3)$ și $(0,3)$ să fie conectate de un drum. Cum densitatea rețelei este cel puțin $D = 1$, vedem că din tripletul $(0,3,4)$ perechea $(3,4)$ trebuie să fie conectate de un drum. În mod similar, atracțiile 0 și 1 trebuie să fie conectate.

În momentul acesta, putem conchide că $t = [1,0,2,3,4]$ este un itinerar de lungime 5, și că nu există deci un itinerar de lungime mai mare ca 5. Așadar procedura `longest_trip` poate returna $[1,0,2,3,4]$.

Considerați alt scenariu în care $N = 4$, $D = 1$, și drumurile dintre atracții sunt precum în figura ce urmează.



Procedura `longest_trip` este apelată în felul următor:

```
longest_trip(4, 1)
```

În acest scenariu, lungimea unui itinerar cel mai lung este 2. Așadar, după câteva apeluri la procedura `are_connected`, procedura `longest_trip` poate întoarce una din $[0, 1]$, $[1, 0]$, $[2, 3]$ sau $[3, 2]$.

Exemplul 2

Subproblema 0 conține un exemplu de caz de test suplimentar cu $N = 256$ atracții. Acest caz de test este inclus în pachetul atașat pe care îl puteti descărca din sistemul de concurs.

Constrângeri

- $3 \leq N \leq 256$
- Suma lui N pentru toate apelurile lui `longest_trip` este cel mult $1\,024$ în fiecare caz de test.
- $1 \leq D \leq 3$

Subprobleme

1. (5 puncte) $D = 3$
2. (10 puncte) $D = 2$
3. (25 puncte) $D = 1$. Fie $l^\star$ lungimea celui mai lung itinerar. Procedura `longest_trip` nu trebuie să întoarcă un itinerar de lungime $l^\star$. Mai degrabă, trebuie să întoarcă un itinerar de lungime cel puțin $\left\lceil \frac{l^\star}{2} \right\rceil$.
4. (60 puncte) $D = 1$

În subtaskul 4 scorul este determinat de numărul apeluri la procedura `are_connected` într-o singură invocare al lui `longest_trip`. Fie q numărul maxim de apeluri pentru toate invocările lui `longest_trip`. Scorul pentru această subproblemă este calculat în concordanță cu tabelul ce urmează:

Condiție	Puncte
$2\,750 < q \leq 32\,640$	20
$550 < q \leq 2\,750$	30
$400 < q \leq 550$	45
$q \leq 400$	60

Dacă în oricare caz de test apelurile la procedura `are_connected` nu se conformează la constrângerile descrise în detaliile de mplementare, sau șirul returnat de `longest_trip` este incorect, scorul soluției voastre va fi 0.

Exemplul de grader

Fie C numărul de scenarii, adică numărul de apeluri la `longest_trip`. Exemplul de grader citeste datele de intrare în următorul format:

- linia 1: C

Descrierile celor C scenarii urmează.

Exemplul de grader citește descrierile scenariilor în următorul format.

- linia 1: $N\ D$
- linia $1 + i$ ($1 \leq i < N$): $U_i[0]\ U_i[1]\ \ldots\ U_i[i-1]$

Aici, fiecaere U_i ($1 \leq i < N$) este un șir de lungime i, descriind care perechi de atracții sunt conectate de câte un drum. Pentru fiecare i și j astfel încât $1 \leq i < N$ și $0 \leq j < i$:

- dacă atracțiile j și i sunt conectate de un drum, atunci valoarea lui $U_i[j]$ ar trebui să fie 1;
- dacă nu există un drum care conectează atracțiile j și i, atunci valoarea lui $U_i[j]$ ar trebui să fie 0.

În fiecare scenariu, înaintea apelării lui `longest_trip`, exemplul de grader verifică dacă densitatea rețelei de drumuri este cel puțin D. Dacă această condiție nu este atinsă, atunci afișază mesajul `Insufficient Density` și închide.

Dacă exemplul de grader detectează o violare a protocolului, atunci outputul exemplului de grader este `Protocol Violation: <MSG>`, unde `<MSG>` este unul dintre următoarele mesaje:

- `invalid array`: intr-un apel la`are_connected`, măcar unul dintre șirurile A sau B
  - este gol, sau
  - conține un element care nu este un întreg între 0 și $n-1$ inclusiv, sau
  - conține același element de cel puțin două ori.
- `non-disjoint arrays`: într-un apel la `are_connected`, șirurile A și B nu sunt distincte.
- `too many calls`: numărul de apeluri făcute la `are_connected` este mai mare de $32\,640$ în timpul invocării curente al lui `longest trip`, sau este mai mare de $150\,000$ în total.
- `too many elements`: numărul total de atracții date lui `are_connected` peste toate apelurile este mai mult decât $1\,500\,000$.

Altfel, fie elementele șirului întors de `longest_trip` într-un scenariu $t[0], t[1], \ldots, t[l-1]$ pentru un t nenegativ. Exemplul de grader afișează trei linii pentru acest scenariu în următorul format:

- linia 1: l

- linia 2: $t[0]$ $t[1]$ $\ldots$ $t[l-1]$
- linia 3: numărul de apeluri la `are_connected` peste acest format.

În cele din urmă, exemplul de grader printează:

- line $1 + 3 \cdot C$: numărul maxim de apeluri la `are_connected` pentru toate apelurile lui `longest_trip`