Najdlhší výlet

Hups... zajtra má byť výlet po parku Ópusztaszer, ale organizátori ho akosi zabudli naplánovať. Nevadí, pomôžeš im ty.

V parku je N zaujímavých lokalít. Očíslujeme si ich od 0 po $N-1$. Niektoré dvojice lokalít sú spojené **cestami**. Všetky cesty sú *obojsmerné*. Medzi každou dvojicou lokalít vedie nanajvýš jedna priama cesta. Organizátori IOI *nevedia* nič o tom, ktoré dvojice lokalít sú prepojené cestou a ktoré nie.

Hovoríme, že **hustota** cestnej siete je **aspoň** δ, ak platí, že každá trojica lokalít má medzi sebou aspoň δ ciest.

Organizátori *poznajú* **kladné** celé číslo D pre ktoré platí, že hustota cestnej siete v parku Ópusztaszer je aspoň D. Inými slovami, pre každé tri lokality (u, v, w) také, že $0 \leq u < v < w < N$, platí, že keď sa pozrieme na dvojice (u, v), (v, w) a (u, w), tak uvidíme, že aspoň D z týchto dvojíc lokalít je prepojených cestou.

Organizátori vedia **zavolať** dispečera v parku Ópusztaszer a získať tým nejaké informácie o cestách medzi lokalitami. Pri každom telefonáte musia dispečerovi nadiktovať dva zoznamy lokalít: $[A[0], \ldots, A[P-1]]$ a $[B[0], \ldots, B[R-1]]$. Lokality v zoznamoch musia byť úplne všetky navzájom rôzne, teda musí platiť:

- $A[i] \neq A[j]$ pre každé i a j také, že $0 \leq i < j < P$;
- $B[i] \neq B[j]$ pre každé i a j také, že $0 \leq i < j < R$;
- $A[i] \neq B[j]$ pre každé i a j také, že $0 \leq i < P$ a $0 \leq j < R$.

Keď dispečer dostane takéto dva zoznamy, odpovie, či existuje nejaká cesta medzi lokalitou v zozname A a lokalitou v zozname B. Presnejšie, dispečer postupne iteruje cez všetky dvojice indexov i a j také, že $0 \leq i < P$ a $0 \leq j < R$. Akonáhle dispečer zistí, že medzi niektorými lokalitami $A[i]$ a $B[j]$ vedie cesta, odpovie `true`. Ak takéto i a j neexistujú, dispečer odpovie `false`.

Výlet dĺžky ℓ je postupnosť *rôznych* lokalít $t[0], t[1], \ldots, t[\ell-1]$, v ktorej pre každé i (od 0 po $\ell-2$ vrátane) platí, že lokality $t[i]$ a $t[i+1]$ sú prepojené cestou.

Výlet dĺžky ℓ voláme **najdlhším výletom** ak neexistuje žiaden výlet dĺžky $\ell+1$.

Tvojou úlohou je pomôcť organizátorom IOI pomocou telefonátov dispečerovi nájsť najdlhší výlet v parku Ópusztaszer.

Detaily implementácie

Implementuj nasledovnú funkciu:

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

- N: počet lokalít v parku.
- D: zaručená minimálna hustota cestnej siete v parku.
- Návratovou hodnotou je pole $t = [t[0], t[1], \ldots, t[\ell - 1]]$ obsahujúce jeden možný najdlhší výlet.
- Počas jedného testu môže testovač túto funkciu postupne zavolať **viackrát**.

Z tvojej funkcie môžeš volať nasledujúcu funkciu testovača:

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

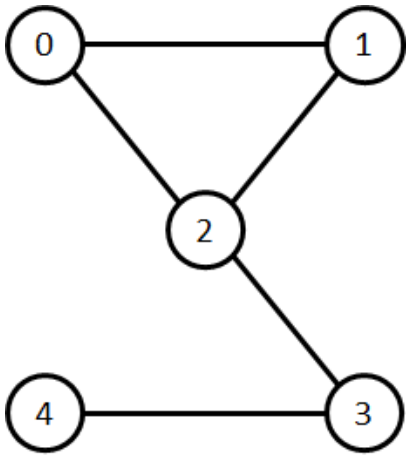
- A: neprázdne pole navzájom rôznych lokalít.
- B: neprázdne pole navzájom rôznych lokalít.
- Polia A a B musia byť disjunktné (t.j. nesmú obsahovať tú istú lokalitu).
- Táto funkcia vráti `true` ak existuje aspoň jedna cesta medzi nejakou lokalitou z A a nejakou lokalitou z B. Inak táto funkcia vráti `false`.
- Z každého volania tvojej funkcie `longest_trip` smieš túto funkciu zavolať nanajvýš $32\,640$-krát.
- Počas celého jedného testu môžeš túto funkciu dokopy zavolať nanajvýš $150\,000$-krát.
- Dokopy pre každý jeden celý test musí platiť, že keď sčítame dĺžky všetkých polí A a B vo všetkých volaniach tejto funkcie, tak výsledný súčet neprekročí $1\,500\,000$.

Testovač **nie je adaptívny**. V každom teste sú vopred pevne zvolené aj všetky hodnoty N a D, aj všetky dvojice lokalít prepojených cestou.

Príklady

Príklad 1

Uvažujme park, v ktorom $N = 5$, $D = 1$ a cesty zodpovedajú nasledovnému obrázku:



Pre tento park by testovač tvoju funkciu `longest_trip` zavolal nasledovne:

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

Následne by napríklad tvoja funkcia mohla volať funkciu `are_connected` takto:

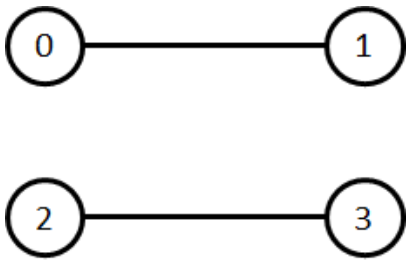
Volanie	Dvojice prepojené cestou	Návratová hodnota
`are_connected([0], [1, 2, 4, 3])`	$(0, 1)$ a $(0, 2)$	`true`
`are_connected([2], [0])`	$(2, 0)$	`true`
`are_connected([2], [3])`	$(2, 3)$	`true`
`are_connected([1, 0], [4, 3])`	žiadne	`false`

Pri štvrtom volaní sme sa dozvedeli, že žiadna z dvojíc $(1, 4)$, $(0, 4)$, $(1, 3)$ a $(0, 3)$ nie je prepojená cestou. Keďže vieme, že cestná sieť má hustotu aspoň $D = 1$, vieme si pre trojicu $(0, 3, 4)$ odvodiť, že dvojica $(3, 4)$ musí byť spojená cestou. Rovnako si vieme odvodiť aj to, že dvojica $(0, 1)$ musí mať medzi sebou cestu.

V tomto okamihu už teda vieme prehlásiť, že $t = [1, 0, 2, 3, 4]$ je platným výletom dĺžky 5. A keďže žiadny dlhší výlet existovať nemôže, toto t predstavuje najdlhší výlet. Tvoja funkcia `longest_trip` by teda mohla vrátiť pole $[1, 0, 2, 3, 4]$.

Príklad 2

Uvažujme park, v ktorom $N = 4$, $D = 1$ a cesty vyzerajú ako na nasledujúcom obrázku:



Volanie tvojej funkcie `longest_trip` bude nasledovné:

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

V tomto parku platí, že najdlhší výlet má dĺžku len 2. Tvoja funkcia by teda mala postupne spraviť niekoľko volaní funkcie `are_connected` a následne vrátiť jeden z nasledujúcich štyroch výletov: $[0, 1]$, $[1, 0]$, $[2, 3]$ alebo $[3, 2]$.

Obmedzenia

- $3 \leq N \leq 256$
- V každom teste platí, že keď sčítame hodnoty N pre všetky volania funkcie `longest_trip`, tak tento súčet neprekročí $1\,024$.
- $1 \leq D \leq 3$

Podúlohy

1. (5 bodov) $D = 3$
2. (10 bodov) $D = 2$
3. (25 bodov) $D = 1$. V tejto podúlohe nemusí tvoja funkcia `longest_trip` nájsť najdlhší možný výlet. Nech $\ell^\star$ označuje dĺžku najdlhšieho výletu. Potom stačí, aby tvoja funkcia vrátila ľubovoľný výlet dĺžky aspoň $\left\lceil \frac{\ell^\star}{2} \right\rceil$.
4. (60 bodov) $D = 1$

V podúlohe 4 navyše platí, že tvoj počet bodov závisí od počtu volaní funkcie `are_connected` počas jedného volania tvojej funkcie `longest_trip`. Nech q je najväčší počet volaní `are_connected` počas jedného volania `longest_trip` (pričom toto maximum berieme cez všetky volania vo všetkých testoch pre túto podúlohu). Tvoje body za túto podúlohu vypočítame z q podľa nasledujúcej tabuľky:

Rozsah hodnôt	Body
$2\,750 < q \leq 32\,640$	20
$550 < q \leq 2\,750$	30
$400 < q \leq 550$	45
$q \leq 400$	60

Akonáhle v ľubovoľnom teste spravíš volanie funkcie `are_connected` s parametrami nespĺňajúcimi požiadavky uvedené v časti Detaily implementácie, a taktiež akonáhle v ľubovoľnom teste ľubovoľné volanie tvojej funkcie `longest_trip` vráti nesprávny výlet, dostaneš za celú príslušnú podúlohu nula bodov.

Ukážkový testovač

Nech C je počet scenárov, ktoré sa majú postupne otestovať počas jedného testu -- teda počet volaní `longest_trip`, ktoré má testovač postupne spraviť.

Ukážkový testovač začne tým, že načíta zo vstupu riadok s hodnotou C:

- riadok 1: C

Na vstupe bude ďalej nasledovať popis C scenárov, postupne jeden za druhým.

Každý scenár má popis v nasledovnom tvare:

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

Každé U_i (pre $1 \leq i < N$) je pole veľkosti i popisujúce, ktoré dvojice lokalít sú prepojené cestou. Presnejšie, pre každé i a j také, že $1 \leq i < N$ a $0 \leq j < i$, platí:

- Ak lokality j a i sú spojené cestou, tak $U_i[j]$ má byť 1;
- ak prepojené nie sú, tak $U_i[j]$ má byť 0.

Pre každý scenár ukážkový testovač začne tým, že skontroluje, či má daná cestná sieť naozaj hustotu aspoň D. Ak nie, vypíše správu `Insufficient Density` a skončí. Ak áno, zavolá tvoju funkciu `longest_trip`.

Ak nastane problém pri komunikácii medzi tvojím programom a ukážkovým testovačom, podá ukážkový testovač správu `Protocol Violation: <MSG>`, kde `<MSG>` je vhodná z nasledujúcich chybových hlášok:

- `invalid array`: pri volaní `are_connected` je aspoň jedno z polí A a B samo o sebe nesprávne (t.j. je prázdne, obsahuje neplatné číslo lokality, alebo obsahuje nejaké číslo lokality viackrát)
- `non-disjoint arrays`: pri volaní `are_connected` sa nejaká lokalita vyskytuje v poli A aj B.
- `too many calls`: bol prekročený niektorý z dvoch limitov na počet volaní funkcie `are_connected`.
- `too many elements`: bol prekročený limit na súčet dĺžok všetkých polí A a B.

Ak všetko úspešne prebehne a tvoja funkcia `longest_trip` vráti nejaké pole, označme si jeho prvky $t[0], t[1], \ldots, t[\ell-1]$ pre nejaké nezáporné ℓ. Ukážkový testovač teraz vypíše na výstup tri riadky v nasledovnom formáte:

- riadok 1: ℓ
- riadok 2: $t[0]$ $t[1]$ $\ldots$ $t[\ell-1]$
- riadok 3: počet volaní `are_connected` počas riešenia tohto scenára

Po dokončení posledného scenára ukážkový testovač vypíše ešte jeden riadok:

- line $1 + 3 \cdot C$: maximum počtov volaní funkcie `are_connected` počas jednotlivých vykonávaní `longest_trip`