Najdłuższa wycieczka

Organizatorzy IOI 2023 są w wielkich tarapatach! Zapomnieli zaplanować jutrzejszą wycieczkę do Ópusztaszer. Może nie jest jednak zbyt późno...

W Ópusztaszer znajduje się N zabytków ponumerowanych od 0 do $N-1$. Niektóre pary tych zabytków są połączone *dwukierunkowymi* **drogami**. Każda para zabytków jest połączona co najwyżej jedną drogą. Organizatorzy *nie wiedzą*, które zabytki są połączone drogami.

Powiemy, że **gęstość** sieci drogowej Ópusztaszer jest równa **co najmniej** δ, jeżeli każda trójka różnych zabytków zawiera co najmniej δ dróg. Innymi słowy, dla każdej trójki zabytków (u, v, w), takiej że $0 \leq u < v < w < N$, spośród par zabytków $(u, v), (v, w)$ oraz (u, w) co najmniej δ z nich jest połączonych drogą.

Organizatorzy *znają* dodatnią liczbę całkowitą D, taką że gęstość sieci drogowej wynosi co najmniej D. Zwróć uwagę na to, że wartość D jest nie większa niż 3.

Organizatorzy mogą zadzwonić do dyspozytora w Ópusztaszer, aby zadać **pytania** dotyczące dróg łączących pewne zabytki. W jednym pytaniu podają dwie niepuste tablice zabytków $[A[0], \ldots, A[P-1]]$ oraz $[B[0], \ldots, B[R-1]]$. Zabytki muszą być parami różne, czyli:

- $A[i] \neq A[j]$ dla wszystkich i oraz j, takich że $0 \leq i < j < P$;
- $B[i] \neq B[j]$ dla wszystkich i oraz j, takich że $0 \leq i < j < R$;
- $A[i] \neq B[j]$ dla wszystkich i oraz j, takich że $0 \leq i < P$ oraz $0 \leq j < R$.

Po każdym zapytaniu dyspozytor iteruje się po wszystkich i oraz j, takich że $0 \leq i < P$ oraz $0 \leq j < R$. Jeśli dla którychś z nich zabytek $A[i]$ jest połączony drogą z zabytkiem $B[j]$, dyspozytor odpowiada `true`. W przeciwnym wypadku dyspozytor odpowiada `false`.

Wycieczka długości l jest ciągiem *różnych* zabytków $t[0], t[1], \ldots, t[l-1]$, gdzie dla każdego i pomiędzy 0 a $l-2$ (włącznie), zabytek $t[i]$ oraz zabytek $t[i+1]$ są połączone drogą. Wycieczka długości l jest **najdłuższą wycieczką** jeśli nie istnieje wycieczka długości $l+1$.

Twoim zadaniem jest pomóc organizatorom i znaleźć najdłuższą wycieczkę w Ópusztaszer przez zadawanie pytań dyspozytorowi.

Szczegóły implementacji

Powinieneś zaimplementować następującą procedurę:

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

- N: liczba zabytków w Ópusztaszer.
- D: gwarantowana minimalna gęstość sieci drogowej w Ópusztaszer.
- Ta procedura powinna zwrócić tablicę $t = [t[0], t[1], \ldots, t[l-1]]$ opisującą najdłuższą wycieczkę.
- Ta procedura może zostać wywołana **wielokrotnie** w pojedynczym przypadku testowym.

Powyższa procedura może wywoływać poniższą procedurę:

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

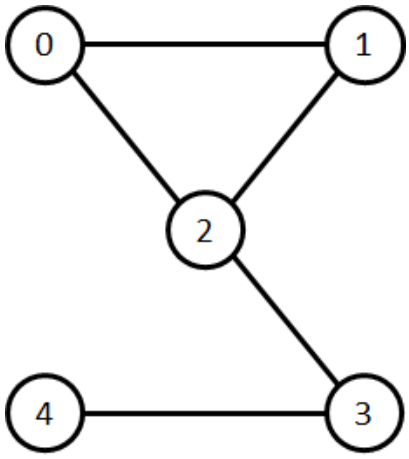
- A: niepusta tablica różnych zabytków.
- B: niepusta tablica różnych zabytków.
- A oraz B muszą być rozłączne.
- Ta procedura zwróci `true`, jeśli istnieje zabytek w A oraz zabytek w B, które są połączone drogą. W przeciwnym wypadku zwróci `false`.
- Ta procedura może zostać wywołana co najwyżej $32\,640$ razy w każdym wywołaniu procedury `longest_trip` oraz co najwyżej $150\,000$ razy sumarycznie.
- We wszystkich wywołaniach tej procedury sumaryczna długość przekazanych jej tablic A oraz B nie może przekroczyć $1\,500\,000$.

Program oceniający **nie jest adaptywny**. Wartości N oraz D, a także pary zabytków połączonych drogami są ustalone przed wywołaniem procedury `longest_trip`.

Przykłady

Przykład 1

Rozważmy scenariusz, gdzie $N = 5$, $D = 1$ oraz połączenia drogowe są jak na rysunku poniżej:



Procedura `longest_trip` jest wywołana w następujący sposób:

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

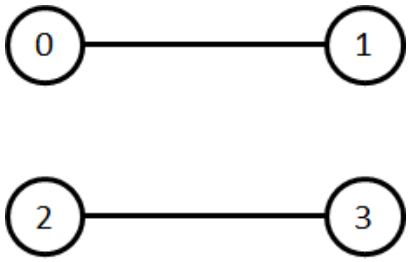
Procedura może wywoływać `are_connected` w następujący sposób.

Wywołanie	Pary połączone drogą	Wynik
`are_connected([0], [1, 2, 4, 3])`	$(0,1)$ oraz $(0,2)$	`true`
`are_connected([2], [0])`	$(2,0)$	`true`
`are_connected([2], [3])`	$(2,3)$	`true`
`are_connected([1, 0], [4, 3])`	brak	`false`

Po czwartym wywołaniu wiemy, że *żadna* z par $(1,4)$, $(0,4)$, $(1,3)$ oraz $(0,3)$ nie jest połączona drogą. Skoro gęstość sieci drogowej wynosi co najmniej $D = 1$ wiemy, że spośród trójki $(0,3,4)$ para $(3,4)$ musi być połączona drogą. Analogicznie, zabytki 0 oraz 1 muszą być połączone.

W tym momencie możemy wywnioskować, że $t = [1,0,2,3,4]$ jest poprawną wycieczką długości 5 oraz nie istnieje wyciecza długości większej niż 5. Zatem, $[1,0,2,3,4]$ jest poprawnym wynikiem procedury `longest_trip`.

Rozważmy inny scenariusz, gdzie $N = 4$, $D = 1$ oraz połączenia drogowe są jak na rysunku poniżej:



Procedura `longest_trip` jest wywołana w następujący sposób:

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

W tym scenariuszu długość najdłuższej wycieczki wynosi 2. Zatem, po kilku wywołaniach procedury `are_connected`, wynikiem procedury `longest_trip` może być $[0,1]$, $[1,0]$, $[2,3]$ lub $[3,2]$.

Przykład 2

Podzadanie 0 zawiera dodatkowy test przykładowy z $N = 256$ zabytkami. Ten test znajduje się w załączonej paczce, którą możesz pobrać z systemu zawodów.

Ograniczenia

- $3 \leq N \leq 256$
- Suma N po wszystkich wywołaniach `longest_trip` nie przekracza $1\,024$.
- $1 \leq D \leq 3$

Podzadania

1. (5 punktów) $D = 3$
2. (10 punktów) $D = 2$
3. (25 punktów) $D = 1$. Niech $l^\star$ oznacza długość najdłuższej wycieczki. Jeśli procedura `longest_trip` zwróci wycieczkę długości co najmniej $\left\lceil \frac{l^\star}{2} \right\rceil$, rozwiązanie zostanie uznane za poprawne.
4. (60 punktów) $D = 1$

W podzadaniu 4 Twój wynik jest uzależniony od liczby wywołań procedury `are_connected` w pojedynczym wywołniu procedury `longest_trip`. Niech q oznacza maksymalną liczbę wywołań `are_connected` spośród wszystkich wywołań `longest_trip` we wszystkich przypadkach testowych tego podzadania. Twój wynik jest obliczany według poniższej tabelki:

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

Jeśli w którymkolwiek przypadku testowym któreś wywołanie procedury `are_connected` nie spełnia ograniczeń opisanych w sekcji Szczegóły implementacji lub tablica zwrócona przez `longest_trip` jest niepoprawna, wynik Twojego zgłoszenia za odpowiadające mu podzadanie wyniesie 0.

Przykładowy program oceniający

Niech C oznacza liczbę scenariuszy $-$ wywołań procedury `longest_trip`. Przykładowy program oceniający wczytuje test ze standardowego wejścia w następującym formacie:

- wiersz 1: C

Następnie występują opisy C scenariuszy.

Przykładowy program oceniający wczytuje opis każdego scenariusza w następującym formacie:

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

Tutaj, każde U_i ($1 \leq i < N$) jest tablicą rozmiaru i opisującą, które pary zabytków są połączone drogą. Dla każdych i oraz j, takich że $1 \leq i < N$ oraz $0 \leq j < i$:

- jeśli zabytki j oraz i są połączone drogą, to wartość $U_i[j]$ jest równa 1;
- jeśli nie istnieje droga łącząca zabytki j oraz i, to wartość $U_i[j]$ jest równa 0.

W każdym scenariuszu, przed wywołaniem `longest_trip`, przykładowy program oceniający sprawdza, czy gęstość podanej sieci drogowej wynosi conajmniej D. Jeśli ten warunek nie jest spełniony, program zwraca komunikat `Insufficient Density` oraz kończy swoje wykonanie.

Jeśli przykładowy program oceniający wykryje naruszenie protokołu komunikacji, zwróci komunikat `Protocol Violation: <MSG>`, gdzie `<MSG>` jest jednym z poniższych komunikatów błędu:

- `invalid array`: w wywołaniu `are_connected`, przynajmniej jedna z tablic A oraz B
  - jest pusta, lub
  - zawiera element, który nie jest liczbą całkowitą z przedziału domkniętego od 0 do $N - 1$ lub
  - zawiera powtarzający się element.
- `non-disjoint arrays`: w wywołaniu `are_connected`, tablice A oraz B nie są rozłączne.
- `too many calls`: liczba wywołań `are_connected` przekracza $32\,640$ w aktualnym wywołaniu `longest trip`, lub sumarycznie przekracza $150\,000$.
- `too many elements`: sumaryczna liczba zabytków przekazanych procedurze `are_connected` we wszystkich wywołaniach przekracza $1\,500\,000$.

W przeciwnym wypadku, oznaczmy elementy tablicy zwróconej przez `longest_trip` jako $t[0], t[1], \ldots, t[l-1]$, gdzie l to nieujemna liczba całkowita. Przykładowy program oceniający wypisze trzy wiersze dla każdego scenariusza w następującym formacie:

- wiersz 1: l
- wiersz 2: $t[0]\ t[1]\ \ldots\ t[l-1]$
- wiersz 3: liczba wywołań `are_connected` w tym scenariuszu

Na koniec przykładowy program oceniający wypisze:

- wiersz $1 + 3 \cdot C$: największa liczba wywołań `are_connected` we wszystkich wywołaniach `longest_trip`