Camino Más Largo

¡Los organizadores de la IOI 2023 están en un problema grande! Se les olvidó planear el paseo a Ópusztaszer para el dia siguiente. Pero tal vez todavía no es demasiado tarde ...

Hay N atracciones en Ópusztaszer numeradas de 0 a $N-1$. Algunos pares de estas atracciones están conectados por **carreteras** *bidireccionales*. Cada par de atracciones está conectado por a lo sumo una carretera. Los organizadores *no saben* cuáles atracciones están conectadas por carreteras.

Decimos que la **densidad** de una red de carreteras en Ópusztaszer es **al menos** δ si cada 3 atracciones distintas tienen al menos δ carreteras entre ellas. En otras palabras, para cada terna de atracciones (u, v, w) tales que $0 \leq u < v < w < N$, entre los pares de atracciones $(u, v), (v, w)$ y (u, w) al menos δ pares están conectadas por una carretera.

Los organizadores *conocen* un entero positivo D tal que la densidad de la red de carreteras es al menos D. Note que el valor de D no puede ser mayor que 3.

Los organizadores pueden hacer **llamadas** telefónicas al informador de Ópusztaszer para obtener información sobre las conexiones de carreteras entre ciertas atracciones. En cada llamada se debe especificar dos arreglos no vacíos de atracciones $[A[0], \ldots, A[P-1]]$ y $[B[0], \ldots, B[R-1]]$. Las atracciones deben ser distintas dos a dos; es decir,

- $A[i] \neq A[j]$ para cada i y j tales que $0 \leq i < j < P$;
- $B[i] \neq B[j]$ para cada i y j tales que $0 \leq i < j < R$;
- $A[i] \neq B[j]$ para cada i y j tales que $0 \leq i < P$ y $0 \leq j < R$.

Para cada llamada, el informador reporta si hay una carretera conectando una atracción de A con una atracción de B. Más precisamente, el informador itera sobre todos los pares i y j tal que $0 \leq i < P$ y $0 \leq j < R$. Si, para cualquiera de ellas, las atracciones $A[i]$ y $B[j]$ están conectadas por una carretera, el informador devuelve `true`. De lo contrario, el informador devuelve `false`.

Un **camino** de longitud l es una secuencia de atracciones *distintas* $t[0], t[1], \ldots, t[l-1]$, donde para cada i entre 0 y $l-2$, inclusive, las atracciones $t[i]$ y $t[i+1]$ están conectados por una carretera. Un camino de longitud l se llama un **camino más largo** si no existe un camino de longitud al menos $l+1$.

Tu tarea es ayudar a los orgnizadores a encontrar un camino más largo en Ópusztaszer haciendo llamadas al informador.

Detalles de Implementación

Debes implementar la función siguiente:

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

- N: el número de atracciones en Ópusztaszer.
- D: la densidad mínima garantizada para la red de carreteras.
- Esta función debe devolver un arreglo $t = [t[0], t[1], \ldots, t[l-1]]$, representando un camino más largo.
- Esta función puede ser llamada **múltiples veces** en cada caso de prueba.

La función anterior puede hacer llamados a la siguiente función:

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

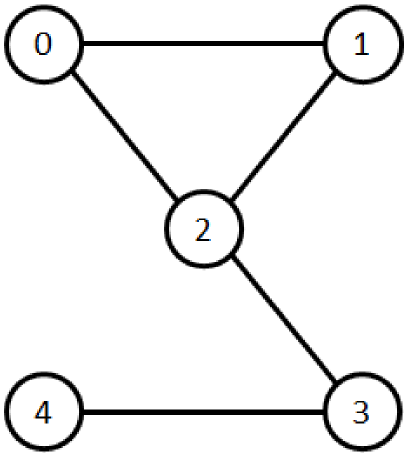
- A: un arreglo no vacío de atracciones distintas.
- B: un arreglo no vacío de atracciones distintas.
- A y B deben ser disjuntos.
- Este procedimiento devuelve `true` si hay una atracción de A y una atracción de B conectadas por una carretera. En otro caso, devuelve `false`.
- Este procedimiento puede ser llamado a lo sumo $32\,640$ veces en cada invocación de `longest_trip`, y a lo sumo $150\,000$ veces en total.
- La longitud total de los arreglos A y B pasados a este procedimiento sobre todas las invocaciones no puede exceder $1\,500\,000$.

El evaluador es **no adaptativo**. Cada envío es evaluado bajo el mismo conjunto de casos de prueba. Es decir, los valores de N y D, así como los pares de atracciones conectadas por carreteras, se fijan antes de hacer un llamada a `longest_trip` dentro de cada caso de prueba.

Ejemplos

Ejemplo 1

Considere un escenario en el cual $N = 5$, $D = 1$, y las carreteras son las mostradas en la figura siguiente:



La función `longest_trip` se llama de la siguiente manera:

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

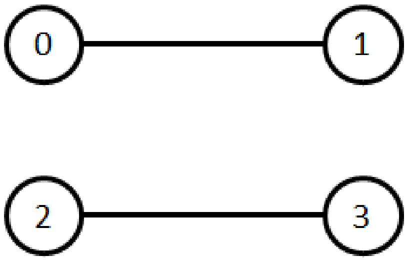
La función puede hacer llamados a `are_connected` de la siguiente manera.

Llamado	Pares conectados por un carretera	Valor devuelto
`are_connected([0], [1, 2, 4, 3])`	$(0, 1)$ y $(0, 2)$	`true`
`are_connected([2], [0])`	$(2, 0)$	`true`
`are_connected([2], [3])`	$(2, 3)$	`true`
`are_connected([1, 0], [4, 3])`	ninguno	`false`

Después de la cuarta llamada, resulta que *ninguno* de los pares $(1, 4)$, $(0, 4)$, $(1, 3)$ y $(0, 3)$ está conectado por una carretera. Como la densidad de la red es al menos $D = 1$, vemos que en la terna $(0, 3, 4)$, el par de atracciones $(3, 4)$ debe estar conectado por una carretera. De manera similar, las atracciones 0 y 1 deben estar conectadas.

En este punto, se puede concluir que $t = [1, 0, 2, 3, 4]$ es un camino de longitud 5, y que no existe un camino de longitud mayor que 5. Por lo tanto, el procedimiento `longest_trip` puede devolver $[1, 0, 2, 3, 4]$.

Considere otro escenario en el cual $N = 4$, $D = 1$, y las carreteras entre las atracciones son como se muestra en la siguiente figura:



La función `longest_trip` se llama de la manera siguiente:

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

En este escenario la longitud de un camino más largo es 2. Por lo tanto, después de unas cuantas llamadas a la función `are_connected`, la función `longest_trip` puede devolver uno de $[0,1]$, $[1,0]$, $[2,3]$ o $[3,2]$.

Ejemplo 2

La subtarea 0 contiene un ejemplo adicional de caso de prueba con $N = 256$ atracciones. Este caso de prueba se incluye en el paquete adjunto que puedes descargar del sistema de competencia.

Restricciones

- $3 \leq N \leq 256$
- La suma de N en todas las llamadas a `longest_trip` no excede $1\,024$ en cada caso de prueba.
- $1 \leq D \leq 3$

Subtareas

1. (5 puntos) $D = 3$
2. (10 puntos) $D = 2$
3. (25 puntos) $D = 1$. Sea $l^\star$ la longitud de un camino más largo. La función `longest_trip` no tiene que devolver un camino de longitud $l^\star$. En vez de eso, debería devolver un camino de longitud al menos $\left\lceil \frac{l^\star}{2} \right\rceil$.
4. (60 puntos) $D = 1$

En la subtarea 4 su puntaje se determina basado en el numero de llamadas a la función `are_connected` en una sola invocación a `longest_trip`. Sea q el número máximo de llamadas en todas las invocaciones a `longest_trip` en todos los casos de prueba de la subtarea. Su puntaje para esta subtarea se calcula de acuerdo a la siguiente tabla:

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

Si en cualquiera de los casos de prueba, los llamados a la función `are_connected` no cumplen con las restricciones decritas en `Detalles de Implementación`, o el arreglo devuelto por `longest_trip` no es correcto, el puntaje de tu solución para esa subtarea será 0.

Evaluador local

Sea C el numero de escenarios; es decir, el numero de llamadas a `longest_trip`. El calficador local lee la entrada en el siguiente formato:

- linea 1: C

Siguen las descripciones de C escenarios.

El evaluador local lee la descripción de cada escenario en el siguiente formato:

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

Aqui, cada U_i ($1 \leq i \leq N$) es un arreglo de longitud i, describiendo qué parejas de atracciones están conectadas por una carretera. Para cada i y j tales que $1 \leq i < N$ y $0 \leq j < i$:

- Si las atracciones j e i están conectadas por una carretera, entonces el valor de $U_i[j]$ debe ser 1;
- Si no hay una carretera conectando las atracciones j e i, entonces el valor de $U_i[j]$ debe ser 0.

En cada escenario, antes de llamar a `longest_trip`, el evaluador local verifica que la densidad de la red de carreteras sea al menos D. Si la condicion no se cumple, imprime el mensaje `Insufficient Density` y termina.

Si el evaluador local detecta una violación a las restricciones imprimirá `Protocol Violation: <MSG>`, donde `<MSG>` es uno de los siguientes mensajes de error:

- `invalid array`: en una llamada a `are_connected`, al menos uno de los arreglos A y B
  - esta vacio, o
  - contiene un elemento que no es un entero entre 0 y $N-1$, inclusive, o
  - contiene el mismo elemento al menos dos veces

- `non-disjoint arrays`: en una llamada a `are_connected`, los arreglos A y B no son disjuntos.
- `too many calls`: el número de llamadas hechos a `are_connected` excede $32\,640$ en la invocacion actual de `longest trip`, o excede $150\,000$ en total.
- `too many elements`: el número total de atracciones enviadas a `are_connected` en todas las llamadas excede $1\,500\,000$.

De lo contrario, sean $t[0], t[1], \ldots, t[l-1]$, los elementos del arreglo devueltos por `longest_trip` en un escenario, para algún l no negativo. El evaluador local imprime tres lineas para este escenario en el siguiente formato:

- linea 1: l
- linea 2: $t[0]\ t[1]\ \ldots\ t[l-1]$
- linea 3: el número de llamadas a `are_connected` en este escenario.

Finalmente, el evaluador local imprime:

- linea $1 + 3 \cdot C$: El máximo número de llamados a `are_connected` sobre todas las llamadas a `longest_trip`