# El viaje más largo

¡Los organizadores de la IOI están en un gran problema! Olvidaron planear el viaje a Ópusztaszer para el día siguiente. Pero tal vez aún no es demasiado tarde...

Hay $N$ monumentos en Ópusztaszer enumerados de $0$ a $N-1$. Algunos pares de estos monumentos son conectados por **caminos** *bidireccionales*. Cada par de monumentos son conectados a lo mucho por un camino. Los organizadores *no conocen* cuales monumentos están conectados por caminos.

Decimos que la **densidad** de la red de caminos en Ópusztaszer es **al menos** $\delta$ si cada $3$ distintos monumentos tienen al menos $\delta$ caminos a través de ellos. En otras palabras, por cada tripleta de monumentos $(u, v, w)$ tal que $0 \leq u < v < w < N$, a través de los pares de monumentos $(u, v), (v, w)$ y $(u, w)$ al menos $\delta$ pares están conectados por un camino.

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

Los organizadores pueden hacer **llamadas telefónicas** al celular del centro de información de Ópusztaszer para obtener información sobre las conexiones de caminos entre monumentos. En cada llamada telefónica, dos arreglos no vacíos de monumentos $[A[0], \ldots, A[P-1]]$ y $[B[0], \ldots, B[R-1]]$ deben ser especificados. Los monumentos deben ser distintos dos a dos, es decir:

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

Para cada llamada, el centro de información reporta si hay un camino conectando un monumento en $A$ y un monumento en $B$. Mas precisamente, el centro de información itera sobre todos los pares $i$ y $j$ tal que $0 \leq i < P$ y $0 \leq j < R$. Si es que, para alguno de estos valores, los monumentos $A[i]$ y $B[j]$ están conectados por un camino, el centro de información retorna `true`. Caso contrario, el centro de información `false`.

Un **recorrido** de longitud $l$ es una secuencia de *distintos* monumentos $t[0], t[1], \ldots, t[l-1]$, donde para todo $i$ entre $0$ y $l-2$, inclusive, el monumento $t[i]$ y el monumento $t[i+1]$ están conectados por un camino. Un recorrido $l$ es llamado un **recorrido más largo** si no existe ningún otro recorrido de longitud al menos $l+1$.

Tu tarea es ayudar a los organizadores a encontrar un recorrido más largo en Ópusztaszer haciendo llamadas al centro de información.

## Detalles de Implementación

Debes implementar la siguiente función:

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

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

La anterior función puede llamar a la siguiente función:

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

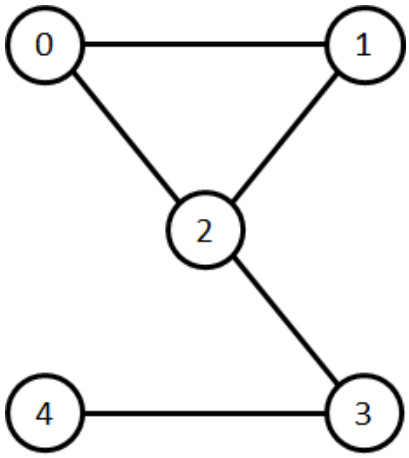
- $A$: un arreglo no vacío de monumentos distintos.
- $B$: un arreglo no vacío de monumentos distintos.
- $A$ y $B$ deben ser disjuntos.
- Esta función retorna `true` si hay un monumento de $A$ y un monumento de $B$ conectados por un camino. En otro caso, retorna `false`.
- Esta función puede ser llamada a lo mucho $32\,640$ veces en cada llamada de `longest_trip`, y a lo sumo $150\,000$ veces en total.
- La longitud total de los arreglos $A$ y $B$ pasados a esta función en todas las llamadas no puede exceder $1\,500\,000$.

El evaluador es **no adaptativo**. Cada envío es evaluado en el mismo conjunto de casos de prueba. Es decir, los valores de $N$ y $D$, así como los pares de monumentos conectados por caminos, se fijan antes de llamar a `longest_trip`.

## Ejemplos

### Ejemplo 1

Considera un escenario en el cual $N = 5$, $D = 1$, y los caminos son los mostrados en la siguiente figura:




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

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

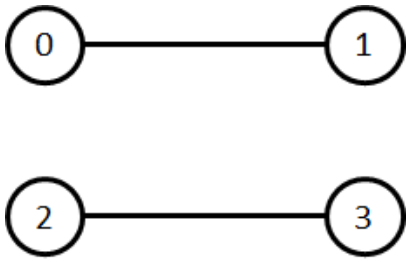
La función puede llamar a `are_connected` de la siguiente forma.

| Llamada | Pares conectados por un camino | Valor retornado |
|---|---|---|
| `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 un camino. Como la densidad de la red es al menos $D = 1$, vemos que en la tripleta $(0,3,4)$, el par de monumentos $(3,4)$ deben estar conectados por un camino. De forma similar, los monumentos $0$ y $1$ deben estar conectados.

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

Considere otro escenario en el cual $N = 4$, $D = 1$, y los caminos entre los monumentos son como se muestra en la siguiente figura:




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

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

En este escenario la longitud de un recorrido más largo es $2$. Por lo tanto, después de unas cuantas llamadas a la función `are_connected`, la función `longest_trip` puede retornar uno de los siguientes: $[0, 1]$, $[1, 0]$, $[2, 3]$ 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 se puede descargar del sistema de competencia.

## Límites

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

## Subtareas

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

En la subtarea 4 el puntaje se determina basado en el número de llamadas a la función `are_connected` en una sola llamada a `longest_trip`. Sea $q$ el número máximo de llamados en todas las llamadas a `longest_trip` en todos los casos de prueba de la subtarea. El 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 descritas en Detalles de Implementación, o el arreglo retornado por `longest_trip` no es correcto, el puntaje de la solución para esta subtarea será de $0$.

## Evaluador de Ejemplo

Sea $C$ el número de escenarios, esto es, el número de llamadas a `longest_trip`. El evaluador de ejemplo lee la entrada en el siguiente formato:

- línea $1$: $C$

Siguen las descripciones de $C$ escenarios.

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

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

Aquí, cada $U_i$ es un arreglo de longitud $i$, describiendo que parejas de monumentos están conectados por un camino. Para cada $i$ y $j$ tales que $1 \leq i < N$ y $0 \leq j < i$:

- Si los monumentos $j$ y $i$ están conectados por un camino, entonces el valor de $U_i[j]$ debe ser $1$;
- Si no hay un camino conectando los monumentos $j$ y $i$, entonces el valor de $U_i[j]$ debe ser $0$.

En cada escenario, antes de ser llamado `longest_trip`, el evaluador de ejemplo verifica que la densidad de la red de carreteras sea al menos $D$. Si la condición no se cumple, imprime el mensaje `Insufficient Density` y termina.

Si el evaluador de ejemplo 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 $B$
  - está vacío, 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 llamados hechos a `are_connected` excede $32\,640$ en la llamada actual a `longest trip`, o excede $150\,000$ en total.
- `too many elements`: El número total de monumentos enviados a `are_connected` en todas las llamadas excede $1\,500\,000$.

En otro caso, 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 de ejemplo imprime tres líneas para este escenario en el siguiente formato:

- línea $1$: $l$
- línea $2$: $t[0]\ t[1]\ \ldots\ t[l-1]$
- línea $3$: La cantidad de llamadas a `are_connected` en este escenario.

Finalmente, el evaluador de ejemplo imprime:

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