La excursión más larga

¡Los organizadores de la IOI 2023 están en un gran aprieto! Se han olvidado de preparar la excursión a Ópusztaszer del próximo día. Pero quizás no es demasiado tarde ...

Hay N puntos de interés en Ópusztaszer indexados desde 0 hasta $N-1$. Algunos pares de estos puntos de interés están conectados por **carreteras** *bidireccionales*. Cada par de puntos de interés estan conectados como mucho por una carretera. Los organizadores *no saben* qué pares de puntos de interés están conectados por carreteras.

Decimos que la **densidad** de una red de carreteras en Ópusztaszer es **como mínimo** δ si para cada 3 puntos de interés distintos tenemos como mínimo δ carreteras entre ellos. En otras palabras, para cada triplete de puntos de interés (u, v, w) tales que $0 \leq u < v < w < N$, entre las parejas de puntos de interés $(u, v), (v, w)$ y (u, w) como mínimo δ parejas están conectados por una carretera.

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

Los organizadores pueden hacer **llamadas** a la centralita de Ópusztaszer para obtener información de las conexiones entre ciertos puntos de interés. En cada llamada, se tienen que especificar dos vectores, no vacíos, de puntos de interes $[A[0], \ldots, A[P-1]]$ y $[B[0], \ldots, B[R-1]]$. Los puntos de interés tienen que ser distintos entre ellos, eso es

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

Para cada llamada, la centralita responde si hay alguna carretera entre los puntos de interés de A y los puntos de interés de B. Más precisamente, la centralita itera por todos los pares i y j tal que $0 \leq i < P$ y $0 \leq j < R$. Si, en alguno de ellos, los puntos de interés $A[i]$ y $B[j]$ están conectados por una carretera, la centralita devuelve `true`. Alternativamente, la centralita devuelve `false`.

Una **excursión** de longitud l es una secuencia de *diferentes* puntos de interés $t[0], t[1], \ldots, t[l-1]$, donde para cada i entre 0 y $l-2$, inclusives, los puntos de interés $t[i]$ y $t[i+1]$ están conectados por una carretera. Una excursión de longitud l se llama **excursión de longitud máxima** si no existe ninguna excursión de longitud como mínimo $l+1$.

Tu tarea es ayudar a los organizadores a encontrar una excursión de longitud máxima en Ópusztaszer haciendo las llamadas a la centralita.

Detalles de implementación

Tienes que implementar la siguiente función:

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

- N: el número de puntos de interés en Ópusztaszer.
- D: la densidad mínima garentida de la red de carreteras.
- La función debe devolver un vector $t = [t[0], t[1], \ldots, t[l-1]]$, representando una excursión de longitud máxima.
- Esta función se puede llamar **varias veces** en cada caso de prueba.

La función anterior puede hacer llamadas a la siguiente función:

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

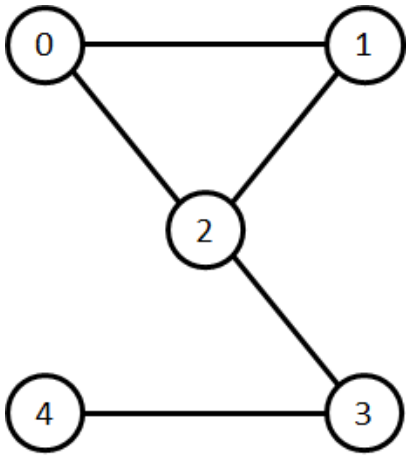
- A: un vector no vacío de puntos de interés distintos.
- B: un vector no vacío de puntos de interés distintos.
- A y B tienen que ser disjuntos.
- La función devuelve `true` si hay un punto de interés de A y un punto de interés de B conectados por una carretera. Si no, devuelve `false`.
- Esta función puede ser llamada como mucho $32\,640$ veces en cada llamada de `longest_trip`, y como mucho $150\,000$ veces en total.
- La suma de las longitudes de los vectores de A y B pasados a esta función sobre todas las llamadas realizadas no puede exceder $1\,500\,000$.

El grader **no es adaptativo**. Cada envío es evaluado en el mismo conjunto de juegos de prueba. Esto es, los valores de N y D, así como los pares de puntos de interés conectados por carreteras, están fijados para cada llamada a `longest_trip` dentro de cada juego de prueba.

Ejemplos

Ejemplo 1

Considera el escenario en el que $N = 5$, $D = 1$, y las conexiones por carretera se pueden ver en la siguiente figura:



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

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

La función puede hacer llamadas a `are_connected` como sigue.

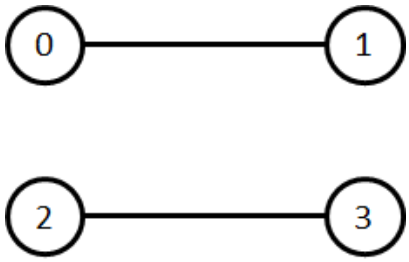
Llamada	Pares conectados por carreteras	Valor devuelto
`are_connected([0], [1, 2, 4, 3])`	$(0, 1)$ y $(0, 2)$	`true`
`are_connected([2], [0])`	$(2, 0)$	`true`
`are_connected([2], [3])`	$(2, 3)$	`true`
`are_connected([1, 0], [4, 3])`	ninguno	`false`

Después de la cuarta llamada, resulta que *ninguno* de los pares $(1, 4)$, $(0, 4)$, $(1, 3)$ y $(0, 3)$ están conectados por una carretera. La densidad de la red es como mínimo $D = 1$, y podemos ver que del triplete $(0, 3, 4)$, el par $(3, 4)$ tienen que estar conectados por una carretera. De manera similar, los puntos de interés de 0 y 1 tienen que estar conectados.

En este momento, podemos concluir que $t = [1, 0, 2, 3, 4]$ es una excursión de longitud 5, y que no existe ninguna excursión de longitud superior a 5.

Así pués, la función `longest_trip` puede devolver $[1, 0, 2, 3, 4]$.

Considera otro escenario en el que $N = 4$, $D = 1$, y que las carreteras entre puntos de interés son como las que se muestran en la siguiente figura:



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

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

En este escenario, la longitud de las excursiones más largas es 2. Así, después de varias llamadas a la función `are_connected`, la función `longest_trip` puede devolver cualquiera de $[0, 1]$, $[1, 0]$, $[2, 3]$ o $[3, 2]$.

Ejemplo 2

La subtarea 0 contiene un ejemplo adicional con un juego de prueba con $N = 256$ puntos de interés. Este caso de prueba está incluido en el paquete adjunto que puedes bajar del CMS.

Restricciones

- $3 \leq N \leq 256$
- La suma de N de todas las llamadas a `longest_trip` no exceden $1\,024$ en cada juego 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 máxima de una excursión. La función `longest_trip` no tiene por qué devolver una excursión de longitud $l^\star$. En vez de eso, puede devolver una excursión de longitud como mínimo $\left\lceil \frac{l^\star}{2} \right\rceil$.
4. (60 puntos) $D = 1$

En la subtarea 4 tu puntuación esta determinada por el número de llamadas a la función `are_connected` en cada llamada a `longest_trip`. Sea q el máximo número de llamadas entre todas las `longest_trip` de todos los juegos de prueba de la subtarea. Tu puntuación para la tarea se calcula en acorde 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 alguno de los juegos de prueba, las llamadas a la función `are_connected` no satisfacen las restricciones descritas en los Detalles de Implementación, o el vector devuelto por `longest_trip` es incorrecto, la puntuación para esta subtarea será de 0.

Grader de ejemplo

Sea C el número de escenarios, eso es, el número de llamadas a `longest_trip`. El grader de ejemplo lee el input con el siguiente formato:

- línea 1: C

Siguen las descripciones de los C escenarios.

El grader 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 ($1 \leq i < N$) es un array de tamaño i, que describe qué pares de puntos de interés están conectados por una carretera. Para cada i y j tal que $1 \leq i < N$ y $0 \leq j < i$:

- si los puntos de interés j e i están conectados por una carretera, entonces el valor de $U_i[j]$ tiene que ser 1;
- si no hay carretera conectando los puntos de interés j e i, entonces el valor de $U_i[j]$ tiene que ser 0.

En cada escenario, antes de llamar `longest_trip`, el grader de ejemplo comprueba que la densidad de la red de carreteras es como mínimo D. Si esta condición no se cumple, escribe el mensaje `Insufficient Density` y acaba.

Si el grader de ejemplo detecta una violación de protocolo, la salida del grader de ejemplo es `Protocol Violation: <MSG>`, donde `<MSG>` es uno de los siguientes mensajes de error:

- `invalid array`: en una llamada a `are_connected`, como mínimo uno de los arrays 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 como mínimo dos veces.

- `non-disjoint arrays`: en una llamada a `are_connected`, arrays A y B no son disjuntos.
- `too many calls`: el número de llamadas 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 puntos de interés pasados a `are_connected` de todas las llamadas excede $1\,500\,000$.

Alternativamente, sean $t[0], t[1], \ldots, t[l-1]$ los elementos del array devueltos por `longest_trip` en un escenario para un número no negativo l. El grader de ejemplo imprime tres líneas para este escenario con el siguiente formato:

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

Finalmente, el grader de ejemplo escribe:

- línea $1 + 3 \cdot C$: el máximo número de llamadas a `are_connected` de todas las llamadas a `longest_trip`