# بلندترین سفر

برگزارکنندگان IOI 2023 در دردسر بزرگی گیر افتادند! آن‌ها فراموش کردند که سفر به Ópusztaszer را برای فردا برنامه ریزی کنند. اما شاید هنوز خیلی دیر نباشد...

در Ópusztaszer، $N$ مکان دیدنی وجود دارد که با شماره‌های $0$ تا $N-1$ شماره‌گذاری شده‌اند. برخی از این مکان‌های دیدنی با **جاده‌های** دو *طرفه* به هم متصل شده‌اند. بین هر دو مکان دیدنی حداکثر یک جاده وجود دارد. برگزارکنندگان **نمی‌دانند** کدام مکان‌ها با جاده به یک‌دیگر متصل هستند.

می‌گوییم **چگالی** یک شبکه از جاده‌ها در Ópusztaszer **حداقل** $\delta$ است اگر به ازای هر سه مکان دیدنی حداقل $\delta$ جاده بین این سه مکان وجود داشته باشد. به عبارت دیگر، به ازای هر سه تایی از مکان‌های دیدنی $(u, v, w)$ به طوری که $0 \leq u < v < w < N$، از بین تمام جفت مکان‌های $(u, v)$، $(v, w)$، و $(u, w)$ حداقل $\delta$ جفت با یک جاده به هم متصل هستند.

برگزارکنندگان عدد مثبت $D$ را می‌*دانند* طوری که چگالی شبکه جاده‌ها حداقل $D$ است. توجه کنید که مقدار $D$ نمی‌تواند بیشتر از $3$ باشد.

برگزارکنندگان می‌توانند با مامور مرکز تلفن Ópusztaszer **تماس بگیرند** تا درباره‌ی ارتباط‌های جاده‌ای بین مکان‌های دیدنی اطلاعات کسب کنند. در هر تماس، دو آرایه‌ی غیرخالی از مکان‌های دیدنی $[A[0], \ldots, A[P-1]]$ و $[B[0], \ldots, B[R-1]]$ باید مشخص شوند. مکان‌های دیدنی باید دو به دو متفاوت باشند، یعنی

- $A[i] \neq A[j]$ به ازای هر $i$ و $j$ طوری که $0 \leq i < j < P$;
- $B[i] \neq B[j]$ به ازای هر $i$ و $j$ طوری که $0 \leq i < j < R$;
- $A[i] \neq B[j]$ به ازای هر $i$ و $j$ طوری که $0 \leq i < P$ and $0 \leq j < R$.

در پاسخ به هر تماس (call)، مامور گزارش می‌دهد که آیا جاده‌ای وجود دارد که یک مکان دیدنی از $A$ و یک مکان دیدنی از $B$ را به یک‌دیگر متصل کند. به طور دقیق‌تر، مامور یکی یکی تمام جفت‌های $i$ و $j$ که $0 \leq i < P$ و $0 \leq j < R$ را بررسی می‌کند. اگر برای هر کدام از آن‌ها، مکان دیدنی $A[i]$ و $B[j]$ با یک جاده به یک‌دیگر متصل باشند مامور 'true' برمی‌گرداند (returns). در غیر این صورت، مامور 'false' برمی‌گرداند.

یک سفر به طول $l$ دنباله‌ای از مکان‌های دیدنی متفاوت $t[0], t[1], \ldots, t[l-1]$ است، طوری که به ازای هر $i$ بین $0$ و $l-2$ (شامل دو سر این بازه) مکان دیدنی $t[i]$ و مکان دیدنی $t[i+1]$ با یک جاده به یک‌دیگر متصل باشند. یک سفر به طول $l$ بلندترین سفر نامیده می‌شود اگر سفری به طول حداقل $l+1$ وجود نداشته باشد.

وظیفه‌ی شما کمک به برگزارکنندگان برای پیدا کردن بلندترین سفر با استفاده از تماس گرفتن با مامور مرکز تلفن است.

## Implementation Details

:You should implement the following procedure

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

- $N$: the number of landmarks at Ópusztaszer.
- $D$: the guaranteed minimum density of the road network.
- This procedure should return an array $t = [t[0], t[1], \ldots, t[l-1]]$, representing a longest trip.
- This procedure may be called **multiple times** in each test case.

:The above procedure can make calls to the following procedure

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

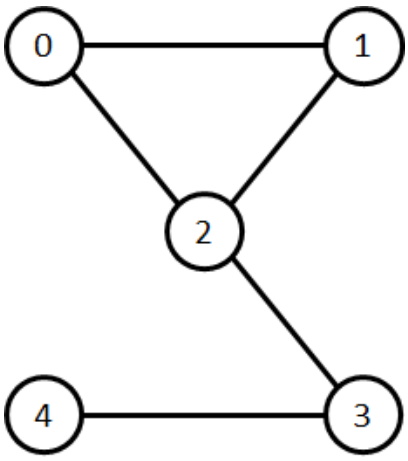
- $A$: a nonempty array of distinct landmarks.
- $B$: a nonempty array of distinct landmarks.
- $A$ and $B$ should be disjoint.
- This procedure returns `true` if there is a landmark from $A$ and a landmark from $B$ connected by a road. Otherwise, it returns `false`.
- This procedure can be called at most $32\,640$ times in each invocation of `longest_trip`, and at most $150\,000$ times in total.
- The total length of arrays $A$ and $B$ passed to this procedure over all of its invocations cannot exceed $1\,500\,000$.

The grader is **not adaptive**. The values of $N$ and $D$, as well as the pairs of landmarks connected by roads, are fixed before a call to `longest_trip` is made.

## Examples

### Example 1

Consider a scenario in which $N = 5$, $D = 1$, and the road connections are as shown in the :following figure



:The procedure `longest_trip` is called in the following way

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

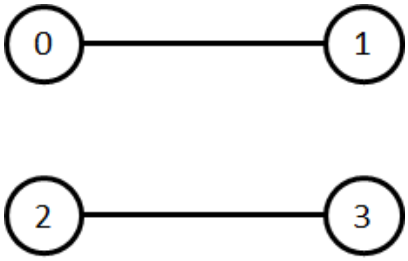
.The procedure may make calls to `are_connected` as follows

| Call | Pairs connected by a road | Return value |
|---|---|---|
| `are_connected([0], [1, 2, 4, 3])` | $(0,1)$ and $(0,2)$ | `true` |
| `are_connected([2], [0])` | $(2,0)$ | `true` |
| `are_connected([2], [3])` | $(2,3)$ | `true` |
| `are_connected([1, 0], [4, 3])` | none | `false` |

After the fourth call, it turns out that *none* of the pairs $(1,4)$, $(0,4)$, $(1,3)$ and $(0,3)$ is connected by a road. As the density of the network is at least $D = 1$, we see that from the triplet $(0,3,4)$, the .pair $(3,4)$ must be connected by a road. Similarly to this, landmarks $0$ and $1$ must be connected

At this point, it can be concluded that $t = [1,0,2,3,4]$ is a trip of length $5$, and that there does not exist a trip of length greater than $5$. Therefore, the procedure `longest_trip` may return .$[1,0,2,3,4]$

Consider another scenario in which $N = 4$, $D = 1$, and the roads between the landmarks are as :shown in the following figure




:The procedure `longest_trip` is called in the following way

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

In this scenario, the length of a longest trip is $2$. Therefore, after a few calls to procedure .$[3,2]$ `are_connected`, the procedure `longest_trip` may return one of $[0,1]$, $[1,0]$, $[2,3]$ or

## Example 2

Subtask 0 contains an additional example test case with $N = 256$ landmarks. This test case is .included in the attachment package that you can download from the contest system

## Constraints

- $3 \leq N \leq 256$
- The sum of $N$ over all calls to `longest_trip` does not exceed $1\,024$.
- $1 \leq D \leq 3$

## Subtasks

1. $D = 3$ (points 5)
2. $D = 2$ (points 10)
3. $D = 1$. Let $l^\star$ denote the length of a longest trip. Procedure `longest_trip` does not have to return a trip of length $l^\star$. Instead, it should return a trip of length at least $\left\lceil \frac{l^\star}{2} \right\rceil$. (points 25)
4. $D = 1$ (points 60)

In subtask 4 your score is determined based on the number of calls to procedure `are_connected` over a single invocation of `longest_trip`. Let $q$ be the maximum number of calls among all invocations of `longest_trip` over every test case of the subtask. Your score for this subtask is calculated according to the following table:

| Condition | Points |
|---|---|
| $2\,750 < q \leq 32\,640$ | 20 |
| $550 < q \leq 2\,750$ | 30 |
| $400 < q \leq 550$ | 45 |
| $q \leq 400$ | 60 |

## Sample Grader

Let $C$ denote the number of scenarios, that is, the number of calls to `longest_trip`. The sample grader reads the input in the following format:

- line 1: $C$

The descriptions of $C$ scenarios follow.

The sample grader reads the description of each scenario in the following format:

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

Here, each $U_i$ ($1 \leq i < N$) is an array of size $i$, describing which pairs of landmarks are connected by a road. For each $i$ and $j$ such that $1 \leq i < N$ and $0 \leq j < i$:

- if landmarks $j$ and $i$ are connected by a road, then the value of $U_i[j]$ should be $1$;
- if there is no road connecting landmarks $j$ and $i$, then the value of $U_i[j]$ should be $0$.

In each scenario, before calling `longest_trip`, the sample grader checks whether the density of the road network is at least $D$. If this condition is not met, it prints the message `Insufficient Density` and terminates.

If the sample grader detects a protocol violation, the output of the sample grader is `Protocol Violation: <MSG>`, where `<MSG>` is one of the following error messages:

- $B$ `invalid array`: in a call to `are_connected`, at least one of arrays $A$ and $B$
  - is empty, or
  - contains an element that is not an integer between $0$ and $N-1$, inclusive, or
  - contains the same element at least twice.
- `non-disjoint arrays`: in a call to `are_connected`, arrays $A$ and $B$ are not disjoint.
- `too many calls`: the number of calls made to `are_connected` exceeds $32\,640$ over the current invocation of `longest trip`, or exceeds $150\,000$ in total.
- `too many elements`: the total number of landmarks passed to `are_connected` over all calls exceeds $1\,500\,000$.

Otherwise, let the elements of the array returned by `longest_trip` in a scenario be $t[0], t[1], \ldots, t[l-1]$ for some nonnegative $l$. The sample grader prints three lines for this scenario in the following format:

- line $1$: $l$
- line $2$: $t[0]\ t[1]\ \ldots\ t[l-1]$
- line $3$: the number of calls to `are_connected` over this scenario

Finally, the sample grader prints:

- line $1 + 3 \cdot C$: the maximum number of calls to `are_connected` over all calls to `longest_trip`