A mais longa viagem

Os organizadores da IOI2023 estão com um grande problema! Eles se esqueceram de planejar a excursão do próximo dia, para Ópusztaszer. Mas talvez não seja tarde demais...

Há N pontos de referência em Ópusztaszer indexados de 0 até $N-1$. Alguns pares desses pontos de referência são conectados por **estradas** *bidirecionais*. Cada par de pontos de referência é conectado por no máximo uma estrada. Os organizadores *não sabem* quais pares de pontos de referência são conectados por estradas.

Dizemos que a **densidade** da rede de estradas em Ópusztaszer é **pelo menos** δ se para cada 3 pontos de referência há pelo menos δ estradas entre eles. Em outras palavras, para cada tripla de pontos de referência (u, v, w) tais que $0 \leq u < v < w < N$, entre os pares de pontos de referência $(u, v), (v, w)$ e (u, w) pelo menos δ pares são conectados por uma estrada.

Os organizadores *conhecem* um inteiro positivo D tal que a densidade da rede de estradas é pelo menos D. Note que o valor de D não pode ser maior do que 3.

Os organizadores podem fazer **chamadas** de telefone para um operador em Ópusztaszer para recolher informação sobre as conexões de estradas entre certos pontos de referência. Em cada chamada, dois vetores não vazios de pontos de referência $[A[0], \ldots, A[P-1]]$ e $[B[0], \ldots, B[R-1]]$ devem ser especificados. Os pontos de referência devem ser distintos par a par, isto é,

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

Para cada chamada, o operador informa se há uma estrada conectando algum ponto de referência no vetor A e algum ponto de referência no vetor B. Mais precisamente, ele itera sobre todos os pares i e j tais que $0 \leq i < P$ e $0 \leq j < R$. Se, para qualquer dos pares, os pontos de referência $A[i]$ e $B[j]$ são conectados por uma estrada, então o operador retorna `true`. Caso contrário, o operador retorna `false`.

Uma **viagem** de comprimento l é uma sequência de pontos de referência *distintos* $t[0], t[1], \ldots, t[l-1]$, onde para cada i entre 0 e $l-2$, inclusive, o ponto de referência $t[i]$ e o ponto de referência $t[i+1]$ são conectados por uma estrada. Uma viagem de comprimento l é chamada uma **viagem mais longa** se não existe uma viagem de comprimento pelo menos $l+1$.

Sua tarefa é ajudar os organizadores a encontrar a viagem mais longa em Ópusztaszer fazendo chamadas ao operador.

Detalhes de Implementação

Você deve implementar o seguinte procedimento:

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

- N: o número de pontos de referência em Ópusztaszer.
- D: a densidade mínima garantida da rede de estradas.
- Este procedimento deve retornar um vetor $t = [t[0], t[1], \ldots, t[l-1]]$, representando a viagem mais longa.
- Este procedimento pode ser chamado **múltiplas vezes** em cada caso de teste.

O procedimento acima pode fazer chamadas para o seguinte procedimento:

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

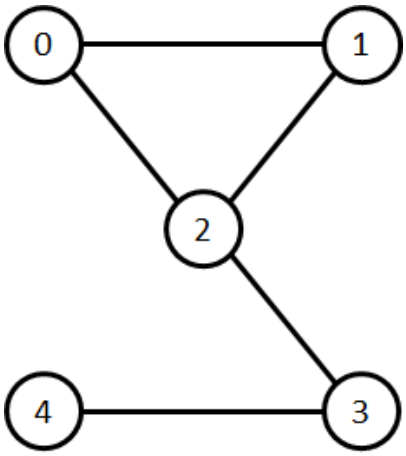
- A: um vetor não vazio de pontos de referência.
- B: um vetor não vazio de pontos de referência.
- A e B devem ser disjuntos.
- Este procedimento retorna `true` se há um ponto de referência em A e um ponto de referência em B conectados por uma estrada. Caso contrário ele retorna `false`.
- Este procedimento pode ser chamado no máximo $32\,640$ vezes em cada invocação de `longest_trip`, e no máximo um total de $150\,000$ vezes.
- O comprimento total dos vetores A e B passados para este procedimento, considerando todas suas invocações não pode exceder $1\,500\,000$.

O grader **não é adaptivo**. Cada submissão é corrigida com o mesmo conjunto de casos de teste. Isto é, os valores de N e D, bem como os pares de pontos de referência conectados por estradas, são fixos para cada chamada a `longest_trip`, em cada caso de teste.

Exemplos

Exemplo 1

Considere um cenário em que $N = 5$, $D = 1$, e as conexões de estradas são como mostrado na seguinte figura:



O procedimento `longest_trip` é chamado da seguinte maneira:

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

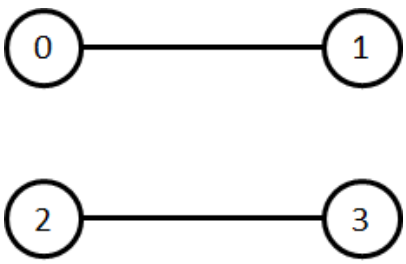
O procedimento pode fazer chamadas para `are_connected` da seguinte forma.

Chamada	Pares conectados por uma estrada	Valor de retorno
`are_connected([0], [1, 2, 4, 3])`	$(0, 1)$ e $(0, 2)$	`true`
`are_connected([2], [0])`	$(2, 0)$	`true`
`are_connected([2], [3])`	$(2, 3)$	`true`
`are_connected([1, 0], [4, 3])`	nenhum	`false`

Após a quarta chamada, *nenhum* dos pares $(1, 4)$, $(0, 4)$, $(1, 3)$ e $(0, 3)$ é conectado por uma estrada. Como a densidade da rede é pelo menos $D = 1$, vemos que da tripla $(0, 3, 4)$, o par $(3, 4)$ é conectado por uma estrada. Similarmente, pontos de referência 0 e 1 são conectados.

Neste ponto, pode-se concluir que $t = [1, 0, 2, 3, 4]$ é uma viagem de comprimento 5, e que não existe uma viagem de comprimento maior do que 5. Portanto, o procedimento `longest_trip` pode retornar $[1, 0, 2, 3, 4]$.

Considere outro cenário em que $N = 4$, $D = 1$, e as estradas entre os pontos de referência são como mostrado na seguinte figura:



O procedimento `longest_trip` é chamado da seguinte maneira:

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

Neste cenário, o comprimento da viagem mais longa é 2. Portanto, após algumas poucas chamadas ao procedimento `are_connected`, o procedimento `longest_trip` pode retornar algum vetor entre $[0, 1]$, $[1, 0]$, $[2, 3]$ e $[3, 2]$.

Exemplo 2

A subtarefa 0 contém um exemplo de caso de teste adicional com $N = 256$ pontos de referência. Este caso de teste é incluído no pacote que você pode baixar do sistema de competição.

Restrições

- $3 \leq N \leq 256$
- A soma dos valores de N considerando todas as chamadas para `longest_trip` não excede $1\,024$.
- $1 \leq D \leq 3$

Subtarefas

1. (5 pontos) $D = 3$
2. (10 pontos) $D = 2$
3. (25 pontos) $D = 1$. Vamos chamar de $l^\star$ o comprimento da viagem mais longa. O procedimento `longest_trip` não precisa retornar uma viagem de comprimento $l^\star$. Ao invés disso, ele deve retornar uma viagem de comprimento pelo menos $\left\lceil \frac{l^\star}{2} \right\rceil$.
4. (60 pontos) $D = 1$

Na subtarefa 4 sua pontuação é determinada com base no número de chamadas ao procedimento `are_connected` considerando uma única invocação de `longest_trip`. Seja q o máximo número de chamadas entre toas as invocações de `longest_trip` considerando cada caso de teste da subtarefa. Sua pontuação para esta subtarefa é calculada de acordo com a seguinte tabela:

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

Se, em qualquer dos casos de teste, as chamadas ao procedimento `are_connected` não obedecer às restrições descritas em Detalhes de Implementação, or o vetor retornado por `longest_trip` estiver incorreto, a pontuação de sua solução para essa subtarefa será 0.

Corretor Exemplo

Seja C o número de cenários, isto é, o número de chamadas a `longest_trip`. O corretor exemplo lê a entrada no seguinte formato:

- linha 1: C

Seguem as descrições de C cenários.

O corretor exemplo lê a descrição de cada cenário no seguinte formato:

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

Aqui, cada U_i ($1 \le i < N$) é um vetor de tamanho i, descrevendo quais pares de pontos de referências são conectados por uma estrada. Para cada i e j tal que $1 \le i < N$ e $0 \le j < i$:

- se o ponto de referência j e i são conectados por uma estrada, então o valor de $U_i[j]$ deve ser 1;
- se não há estrada conectando o ponto de referência j e i, então o valor de $U_i[j]$ deveria ser 0 .

Em cada cenário, antes de chamar `longest_trip`, o corretor exemplo verifica se a densidade da rede de estradas é pelo menos D. Se esta condição não é obedecida, ele imprime a mensagem `Insufficient Density` e termina.

Se o corretor exemplo detecta uma violação de protocolo, a saída do corretor exemplo é `Protocol Violation: <MSG>`, onde `<MSG>` é uma das seguines mensagens:

- `invalid array`: em uma chamada a `are_connected`, pelo menos um dos vetores A e B
  - é vazio, ou
  - contém um elemento que não é um inteiro entre 0 e $N-1$, inclusive, ou
  - contém o mesmo elemento ao menos duas vezes.
- `non-disjoint arrays`: em um chamada a `are_connected`, os vetores A e B não são disjuntos.
- `too many calls`: o número de chamadas feitas a `are_connected` excede $32\,640$ considerando a invocação corrente de `longest trip`, ou excede $150\,000$ no total.
- `too many elements`: o número total de pontos de referência passados para `are_connected` considerando todas as chamadas excede $1\,500\,000$.

Caso contrário, considere que o número de elementos do vetor retornado por`longest_trip` em um cenário seja $t[0], t[1], \ldots, t[l-1]$ para algum l não negativo. O corretor exemplo imprime três linhas para este cenário, no seguinte formato:

- linha 1: l
- linha 2: $t[0]\ t[1]\ \ldots\ t[l-1]$
- linha 3: o número de chamadas a `are_connected` neste cenário

Finalmente, o corretor exemplo imprime:

- linha $1 + 3 \cdot C$: o número máximo de chamadas a `are_connected` considerando todas as chamadas para `longest_trip`