Longest Trip

Os organizadores das IOI 2023 estão em apuros! Esqueceram-se de planear a viagem a Ópusztaszer para o dia seguinte. Mas talvez não seja ainda tarde demais ...

Existem N pontos de interesse em Ópusztaszer, indexados de 0 a $N-1$. Alguns pares desses pontos de interesse estão ligados por **estradas** *bidirecionais*. Cada par de pontos de interesse está ligado por no máximo uma estrada. Os organizadores *não sabem* quais os pontos de interesse que estão ligados por estradas.

Dizemos que a **densidade** da rede de estradas em Ópusztaszer é **pelo menos** δ se cada 3 pontos de interesse distintos tiverem pelo menos δ estradas entre eles. Por outras palavras, para cada trio de pontos de interesse (u, v, w) tais que $0 \leq u < v < w < N$, entre os pares de pontos de interesse $(u, v), (v, w)$ e (u, w) pelo menos δ pares estão ligados por uma estrada.

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

Os organizadores podem fazer **chamadas** para um operador telefónico em Ópusztaszer para colecionar informações sobre as ligações rodoviárias entre determinados pontos de interesse. Em cada chamada, dois arrays não vazios de pontos de interesse $[A[0], \ldots, A[P-1]]$ e $[B[0], \ldots, B[R-1]]$ devem ser especificados. Os pontos de interesse devem ser distintos dois a dois, ou seja,

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

Para cada chamada, o operador informa se existe uma estrada a ligar um ponto de interesse de A a um ponto de interesse de B. Mais precisamente, o operador itera entre todos os pares i e j tais que $0 \leq i < P$ e $0 \leq j < R$. Se, para algum deles, os pontos de interesse $A[i]$ e $B[j]$ estão ligados por uma estrada, o operador devolve `true`. Caso contrário, o operador devolve `false`.

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

A tua tarefa é ajudar os organizadores a encontrar uma viagem mais longa em Ópusztaszer fazendo chamadas para o operador.

Detalhes de Implementação

Deves implementar a seguinte função:

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

- N: a quantidade de pontos de interesse em Ópusztaszer.
- D: a densidade mínima garantida da rede de estradas.
- Esta função devolver um array $t = [t[0], t[1], \ldots, t[l-1]]$, representando a viagem mais longa.
- Esta função pode ser chamada **várias vezes** em cada caso de teste.

A função atrás indicada pode fazer chamadas à seguinte função:

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

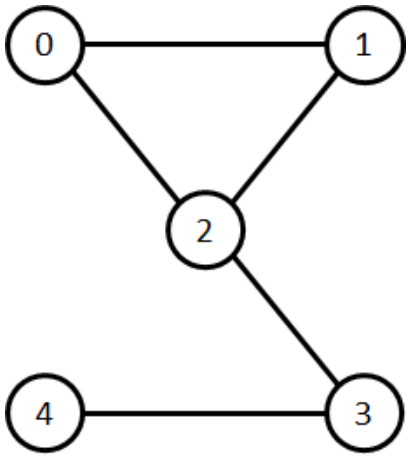
- A: um array não vazio de pontos de interesse distintos.
- B: um array não vazio de pontos de interesse distintos.
- A e B devem ser disjuntos.
- Esta função devolve `true` se existir um ponto de interesse de A e um ponto de interesse de B ligados por uma estrada. Caso contrário, devolve `false`.
- Esta função pode ser chamada no máximo $32\,640$ vezes em cada chamada a `longest_trip`, e no máximo $150\,000$ vezes no total.
- O tamanho total dos arrays A e B passados a esta função em todas as suas chamadas não pode exceder $1\,500\,000$.

O avaliador é **não adaptativo**. Cada submissão é avaliada no mesmo conjunto de casos de teste. Isto é, os valores de N e D, bem como os pares de pontos de interesse ligados por estradas, estão fixados antes de cada chamada a `longest_trip` dentro de cada caso de teste.

Exemplos

Exemplo 1

Considera um cenário onde $N = 5$, $D = 1$ e as ligações rodoviárias são as mostradas na figura seguinte:



A chamada a `longest_trip` é feita da seguinte maneira:

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

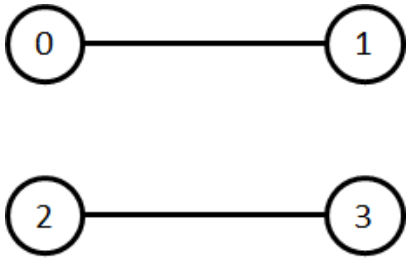
A função poderia fazer chamadas a `are_connected` da seguinte maneira:

Chamada	Pares ligados por uma estrada	Valor devolvido
`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])`	none	`false`

Após a quarta chamada, verifica-se que *nenhum* dos pares $(1,4)$, $(0,4)$, $(1,3)$ e $(0,3)$ está ligado por uma estrada. Como a densidade da rede é de pelo menos $D = 1$, vemos que do trio $(0,3,4)$, o par $(3,4)$ tem de estar ligado por uma estrada. Da mesma forma, os pontos de interesse 0 e 1 devem estar ligados.

Neste altura, pode concluir-se que $t = [1,0,2,3,4]$ é uma viagem de tamanho 5, e que não existe nenhuma viagem de tamanho superior a 5. Portanto, a função `longest_trip` pode devolver $[1,0,2,3,4]$.

Considera outro cenário onde $N = 4$, $D = 1$ e as estradas entre os pontos de interesse são as mostradas na figura seguinte:



A chamada a `longest_trip` é feita da seguinte maneira:

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

Neste cenário, o tamanho de uma viagem mais longa é 2. Portanto, depois de algumas chamadas a `are_connected`, a função `longest_trip` pode devolver um entre $[0, 1]$, $[1, 0]$, $[2, 3]$ ou $[3, 2]$.

Exemplo 2

A subtarefa 0 contém um exemplo de caso de teste adicional com $N = 256$ pontos de interesse. Este caso de teste está incluído no ficheiro em anexo que podes descarregar do sistema de avaliação.

Restrições

- $3 \leq N \leq 256$
- A soma de todos os N entre todas as chamadas a `longest_trip` não excede $1\,024$ em cada caso de teste.
- $1 \leq D \leq 3$

Subtarefas

1. (5 pontos) $D = 3$
2. (10 pontos) $D = 2$
3. (25 pontos) $D = 1$. Seja $l^\star$ o tamanho da viagem mais longa. A função `longest_trip` não tem de devolver uma viagem de tamanho $l^\star$. Ao invés, deve devolver uma viagem com um tamanho de pelo menos $\left\lceil \frac{l^\star}{2} \right\rceil$.
4. (60 pontos) $D = 1$

Na subtarefa 4, a tua pontuação é determinada com base no número de chamadas à função `are_connected` numa única chamda a `longest_trip`. Seja q o número máximo de chamadas entre todas as chamadas a `longest_trip` em cada caso de teste da subtarefa. A tua pontuação para esta subtarefa é calculada de acordo com a tabela a seguir:

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 um dos casos de teste, as chamadas à função `are_connected` não estiverem em conformidade com as restrições descritas nos Detalhes de Implementação, ou o array devolvido por `longest_trip` estiver incorreto, a pontuação da tua solução para aquela subtarefa será 0.

Avaliador Exemplo

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

- linha 1: C

Seguem-se as descrições dos C cenários.

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

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

Aqui, cada U_i ($1 \leq i < N$) é um array de tamanho i, descrevendo quais pares de pontos de interesse estão ligados por uma estrada. Para cada i e j tais que $1 \leq i < N$ e $0 \leq j < i$:

- se os pontos de interesse j e i estão ligados por uma estrada, então o valor de $U_i[j]$ deve ser 1;
- se não existe uma estrada ligada os pontos de interesse j e i, então o valor de $U_i[j]$ deve ser 0.

Em cada cenário, antes de chamar `longest_trip`, o avaliador exemplo verifica se a densidade da rede de estradas é pelo menos D. Se esta condição não for válida, a mensagem `Insufficient Density` será escrita e o avaliador termina a sua execução.

Se o avaliador exemplo detetar uma violação de protocolo, o output será `Protocol Violation: <MSG>`, onde `<MSG>` é uma das seguintes mensagens de erro:

- `invalid array`: numa chamada a `are_connected`, pelo menos um dos arrays A e B
  - está vazio, ou
  - contém um elemento não não é um inteiro entre 0 e $N-1$, inclusive, ou
  - contém o mesmo elemento pelo menos duas vezes.
- `non-disjoint arrays`: numa chamada a `are_connected`, os arrays A and B não são disjuntos.
- `too many calls`: o número de chamadas a `are_connected` excede $32\,640$ na chamada atual a `longest trip`, ou excede $150\,000$ no total.
- `too many elements`: o número total de pontos de interesse passados a `are_connected` em todas as chamadas excede $1\,500\,000$.

Caso contrário, sejam $t[0], t[1], \ldots, t[l-1]$ para algum número l não negativo os elementos do array devolvido por `longest_trip` num cenário. O avaliador exemplo escreve 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 avaliador exemplo escreve:

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