El acertijo de la Esfinge

La gran Esfinge tiene un acertijo para ti. Te es dado un grafo con N vértices. Los vértices están numerados de 0 a $N-1$. Hay M aristas en el grafo, numeradas de 0 a $M-1$. Cada arista conecta un par de vértices distintos y es bidireccional. Más en específico, para cada j de 0 a $M-1$, inclusive, la arista j conecta a los vértices $X[j]$ y $Y[j]$. Existe a lo más una arista conectando cualesquiera dos vértices. Dos vértices se les denomina como **adyacentes** sí están conectados por una arista.

Una secuencia de vértices $v_0, v_1, \ldots, v_k$ (para $k \geq 0$) es llamada un **camino** sí para dos vértices consecutivos v_l y v_{l+1} (para toda l tal que $0 \leq l < k$) son adyacentes. Decimos que un camino $v_0, v_1, \ldots, v_k$ **conecta** vértices v_0 y v_k. En el grafo que te es dado, cada par de vértices está conectado por algún camino.

Hay $N+1$ colores, enumerados de 0 a N. El color N es especial y es llamado el **color de la Esfinge**. A cada vértice se le es asignado un color. Más en específico, el vértice i ($0 \leq i < N$) tiene color $C[i]$. Varios vértices pueden tener el mismo color y puede haber colores a los cuales no se les asigna ningún vértice. Ningún vértice tiene el color de la Esfinge, es decir, $0 \leq C[i] < N$ ($0 \leq i < N$).

Un camino $v_0, v_1, \ldots, v_k$ (para $k \geq 0$) es llamado **monocromático** sí todos sus vértices tienen el mismo color, es decir, $C[v_l] = C[v_{l+1}]$ (para toda l tal que $0 \leq l < k$). Adicionalmente, decimos que los vértices p y q ($0 \leq p < N$, $0 \leq q < N$) están en la misma **componente monocromática** y sí solo sí están conectados por un camino monocromático.

Tú sabes cuales son los vértices y las aristas, pero no sabes el color de cada vértice. Quieres encontrar el color de los vértices, realizando **experimentos de coloración**.

En un experimento de coloración, puedes colorear arbitrariamente los vértices que quieras. Más en específico, para realizar un experimento de coloración primero eliges un arreglo E de tamaño N, donde para toda i ($0 \leq i < N$), $E[i]$ tiene un valor entre -1 y N **inclusive**. Después, el color de cada vértice i se vuelve $S[i]$, donde el valor de $S[i]$ es:

- $C[i]$, es decir, el color original de i, si $E[i] = -1$, o
- $E[i]$, de cualquier otra forma.

Esto significa que puedes usar el color de la Esfinge en tu coloración.

Finalmente, la gran Esfinge anuncia el número de componentes monocromáticas en el grafo, después de colorear cada vértice i al color $S[i]$ ($0 \leq i < N$). El nuevo coloreo se aplica únicamente durante este experimento, entonces **los colores de todos los vértices regresan al color original después que el experimento termine**.

Tu tarea es identificar los colores de los vértices en el grafo haciendo a lo más $2\,750$ experimentos de coloración. Puedes recibir un puntaje parcial sí determinas correctamente para cada par de vértices adyacentes, sí tienen el mismo color.

Detalles de implementación

Debes implementar la siguiente función.

```
std::vector<int> find_colours(int N,
    std::vector<int> X, std::vector<int> Y)
```

- N: el número de vértices en el grafo.
- X y Y son arreglos de tamaño M describiendo las aristas.
- Esta función debe regresar un arreglo G de tamaño N, representando los colores de los vértices en el grafo.
- Esta función la va a llamar el evaluador exactamente una vez por cada caso de prueba.

Tu función puede hacer llamadas a la siguiente función para realizar los experimentos de coloración:

```
int perform_experiment(std::vector<int> E)
```

- E es un arreglo de tamaño N especificando como deben de colorearse los vértices.
- Esta función regresa la cantidad de componentes monocromáticas después del coloreo de acuerdo a E.
- Esta función la puedes llamar a lo más $2\,750$ veces.

El evaluador **no es adaptativo**, es decir, los colores de los vértices se fijan antes de la llamada a `find_colours`.

Límites

- $2 \leq N \leq 250$
- $N - 1 \leq M \leq \frac{N \cdot (N-1)}{2}$
- $0 \leq X[j] < Y[j] < N$ para toda j tal que $0 \leq j < M$.
- $X[j] \neq X[k]$ or $Y[j] \neq Y[k]$ para toda j y k tal que $0 \leq j < k < M$.
- Cada par de vértices está conectado por un camino.
- $0 \leq C[i] < N$ para toda i tal que $0 \leq i < N$.

Subtareas

Subtarea	Puntaje	Consideraciones adicionales
1	3	$N = 2$
2	7	$N \leq 50$
3	33	El grafo es un camino: $M = N - 1$ y los vértices j y $j + 1$ son adyacentes ($0 \leq j < M$).
4	21	El grafo es completo: $M = \frac{N \cdot (N-1)}{2}$ y cualesquiera dos vértices son adyacentes.
5	36	Sin consideraciones adicionales.

En cada subtarea puedes obtener un puntaje parcial si tu programa determina correctamente para cada par de vértices adyacentes si tienen el mismo color.

Más precisamente, obtienes el puntaje total de la subtarea sí en todos sus casos de prueba, el arreglo G regresado por `find_colours` es exactamente el mismo que C (es decir $G[i] = C[i]$ para toda i tal que $0 \leq i < N$). De otra forma, obtienes 50% del puntaje de la subtarea sí se cumplen las siguientes condiciones en todos sus casos de prueba:

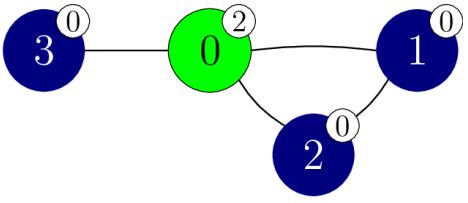
- $0 \leq G[i] < N$ para toda i tal que $0 \leq i < N$;
- Para toda j tal que $0 \leq j < M$:
  - $G[X[j]] = G[Y[j]]$ sí y solo sí $C[X[j]] = C[Y[j]]$.

Ejemplo

Considera la siguiente llamada a tu función.

```
find_colours(4, [0, 1, 0, 0], [1, 2, 2, 3])
```

Para este ejemplo, supón que los colores (ocultos) de los vértices son $C = [2, 0, 0, 0]$. Este ejemplo se muestra en la siguiente imagen. A los colores se les representa adicionalmente como el numerito blanco.



La función puede llamar a `perform_experiment` de la siguiente manera.

```
perform_experiment([-1, -1, -1, -1])
```

En esta llamada, ningún vértice es coloreado, por lo que todos los vértices permanecen en su forma original.

Considera el vértice 1 y vértice 2. Ambos tienen el color 0 y el camino $1, 2$ es un camino monocromático. Por lo tanto, ambos vértices 1 and 2 están en la misma componente monocromática.

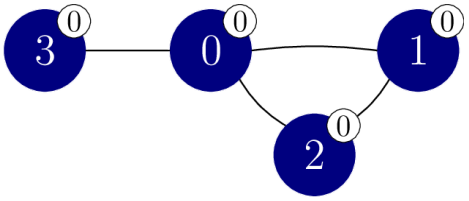
Considera el vértice 1 y el vértice 3. Pese a que ambos tengan el color 0, están en diferentes componentes monocromáticas debido a que no existe un camino monocromático entre ellos.

Al final, hay 3 componentes monocromáticas, con vértices $\\{0\\}$, $\\{1, 2\\}$, y $\\{3\\}$. Entonces, esta llamada regresa 3.

Después, la función puede llamar a `perform_experiment` de la siguiente manera.

```
perform_experiment([0, -1, -1, -1])
```

En esta llamada, solo el vértice 0 es coloreado a 0, lo que resulta en la coloración de la siguiente figura.

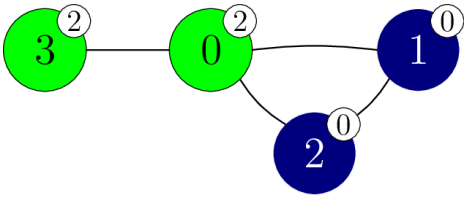


Esta llamada regresa 1, debido a que todos los vértices pertenecen a la misma componente monocromática. Ahora podemos deducir que los vértices 1, 2, y 3 tienen color 0.

La función puede ahora llamar a `perform_experiment` de la siguiente manera.

```
perform_experiment([-1, -1, -1, 2])
```

En esta llamada, el vértice 3 es coloreado a 2, lo que resulta en la coloración de la siguiente figura.



Esta llamada regresa 2, debido a que hay 2 componentes monocromáticas, con vértices $\{0, 3\}$ y $\{1, 2\}$ respectivamente. Podemos deducir que el vértice 0 tiene color 2.

La función `find_colours` regresa entonces el arreglo $[2, 0, 0, 0]$. Dado que $C = [2, 0, 0, 0]$, se obtiene el puntaje completo.

Es importante notar que existen varias respuestas posibles, para las cuales otorgan 50% del puntaje de la subtarea, por ejemplo $[1, 2, 2, 2]$ o $[1, 2, 2, 3]$.

Evaluador de ejemplo

Formato de entrada:

```
N  M
C[0]  C[1] ... C[N-1]
X[0]  Y[0]
X[1]  Y[1]
...
X[M-1]  Y[M-1]
```

Formato de salida:

```
L  Q
G[0]  G[1] ... G[L-1]
```

Donde L es el tamaño del arreglo G regresado por `find_colours`, y Q es el número de llamadas a `perform_experiment`.