Conectando Súper-árboles (supertrees)

Los Jardines de la Bahía es un gran parque natural en Singapur. En el parque hay n torres, conocidas como súper-árboles. Estas torres están etiquetadas del 0 al $n-1$. Nos gustaría construir un conjunto de **cero o más** puentes. Cada puente conecta un par de torres distintas y pueden ser atravesados en **cualquier** dirección. Dos puentes no deben conectar el mismo par de torres.

Un camino de la torre x a la torre y es una secuencia de una o más torres tal que:

- el primer elemento de la secuencia es x,
- el último elemento de la secuencia es y,
- todos los elementos de la secuencia son **distintos**, y
- cada dos elementos consecutivos (torres) en la secuencia están conectados por un puente.

Nota que por definición hay exactamente un camino desde una torre a sí misma y el número de caminos diferentes desde la torre i a la torre j es el mismo que el número de caminos diferentes desde la torre j a la torre i.

El arquitecto líder a cargo del diseño desea que los puentes sean construidos tales que para todo $0 \leq i, j \leq n-1$ hayan exactamente $p[i][j]$ caminos diferentes desde la torre i a la torre j, donde $0 \leq p[i][j] \leq 3$.

Construye un conjunto de puentes que satisfagan los requerimientos de la arquitectura, o determina que es imposible.

Detalles de implementación

Debes implementar el siguiente procedimiento:

```
int construct(int[][] p)
```

- p: un arreglo $n \times n$ representando los requerimientos de la arquitectura.
- Si una construcción es posible, este procedimiento debe hacer exactamente una llamada a `build` (ver debajo) para reportar la construcción, después de lo cual debe retornar 1.
- Si no, el procedimiento debe retornar 0 sin hacer ninguna llamada a `build`.
- Este procedimiento es llamado exactamente una vez.

El procedimiento `build` es definido como sigue:

```
void build(int[][] b)
```

- b: un arreglo $n \times n$, con $b[i][j] = 1$ si hay un puente conectando la torres i y la torre j, o $b[i][j] = 0$ si no.
- Nota que el arreglo debe satisfacer $b[i][j] = b[j][i]$ para todo $0 \leq i, j \leq n - 1$ y $b[i][i] = 0$ para todo $0 \leq i \leq n - 1$.

Ejemplos

Ejemplo 1

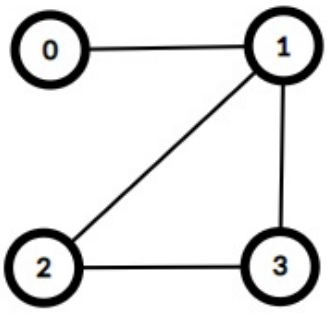
Considera la siguiente llamada:

```
construct([[1, 1, 2, 2], [1, 1, 2, 2], [2, 2, 1, 2], [2, 2, 2, 1]])
```

Esto significa que debe haber exactamente un camino desde la torre 0 a la torre 1. Para todos los otros pares de torres (x, y), tales que $0 \leq x < y \leq 3$, debe haber exactamente dos caminos desde la torre x a la torre y. Esto se puede lograr con 4 puentes, conectando los pares de torres $(0, 1)$, $(1, 2)$, $(1, 3)$ y $(2, 3)$.

Para reportar esta solución, el procedimiento `construct` debe hacer la siguiente llamada:

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



Debe entonces retornar 1.

En este caso, hay múltiples construcciones que satisfacen los requerimientos, todas las cuales serían consideradas como correctas.

Ejemplo 2

Considera la siguiente llamada:

```
construct([[1, 0], [0, 1]])
```

Esto significa que no debe haber manera de viajar entre las dos torres. Esto solo se puede satisfacer no teniendo puentes.

Por lo tanto, el procedimiento `construct` debe hacer la siguiente llamada:

- `build([[0, 0], [0, 0]])`

Después de la cual, el procedimiento `construct` debe retornar 1.

Ejemplo 3

Considere la siguiente llamada:

```
construct([[1, 3], [3, 1]])
```

Esto significa que debe haber exactamente 3 caminos desde la torre 0 a la torre 1. Este conjunto de requerimientos no se puede satisfacer. Así, el procedimiento `construct` debe retornar 0 sin hacer ninguna llamada a `build`.

Restricciones

- $1 \leq n \leq 1000$
- $p[i][i] = 1$ (para todo $0 \leq i \leq n-1$)
- $p[i][j] = p[j][i]$ (para todo $0 \leq i, j \leq n-1$)
- $0 \leq p[i][j] \leq 3$ (para todo $0 \leq i, j \leq n-1$)

Subtareas

1. (11 puntos) $p[i][j] = 1$ (para todo $0 \leq i, j \leq n-1$)
2. (10 puntos) $p[i][j] = 0$ o 1 (para todo $0 \leq i, j \leq n-1$)
3. (19 puntos) $p[i][j] = 0$ o 2 (para todo $i \neq j, 0 \leq i, j \leq n-1$)
4. (35 puntos) $0 \leq p[i][j] \leq 2$ (para todo $0 \leq i, j \leq n-1$) y hay al menos una construcción que satisface los requerimientos.
5. (21 puntos) $0 \leq p[i][j] \leq 2$ (para todo $0 \leq i, j \leq n-1$)
6. (4 puntos) Sin restricciones adicionales.

Evaluador de ejemplo

El evaluador de ejemplo lee la entrada en el siguiente formato:

- línea 1: n
- línea $2+i$ ($0 \leq i \leq n-1$): $p[i][0]$ $p[i][1]$ $\ldots$ $p[i][n-1]$

La salida del evaluador de ejemplo está en el siguiente formato:

- línea 1: el valor retornado de `construct`.

Si el valor retornado de `construct` es 1, el evaluador de ejemplo imprime adicionalmente:

- línea $2 + i$ ($0 \leq i \leq n - 1$): $b[i][0]$ $b[i][1]$ $\ldots$ $b[i][n - 1]$