Conectando los Jardines de la Bahía (supertrees)

Jardines de la Bahía es un extenso parque natural en Singapur. En el parque hay n torres, conocidas como supertrees. Estas torres están numeradas de 0 a $n-1$. Nos gustaría construir un conjunto de **cero o más** puentes. Cada puente conecta un par de torres distintas y puede ser recorrido en **cualquier** dirección. Dos puentes nunca conectarán el mismo par de torres.

Un camino de la torre x a la torre y es una secuencia de una o más torres de tal forma 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.

Puedes notar que por definición hay exactamente un camino de una torre hacia sí misma y el número de caminos distintos desde la torre i hacia la torre j es igual a la cantidad de caminos distintos desde la torre j hacia la torre i.

El arquitecto líder encargado del diseño desea construir los puentes de tal manera que para todo $0 \le i, j \le n-1$ hayan exactamente $p[i][j]$ caminos diferentes de la torre i a la torre j, donde $0 \le p[i][j] \le 3$.

Construye un conjunto de puentes que satisfagan los requerimientos del arquitecto, o determina que es imposible.

Detalles de implementación

Debes implementar la siguiente función:

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

- p: un arreglo de tamaño $n \times n$ representando los requerimientos del arquitecto.
- Si alguna construcción es posible, esta función debe llamar exactamente una vez a `build` (ver abajo) para reportar la construcción. Después, debe regresar 1.
- De lo contrario, la función debe regresar 0 sin hacer ninguna llamada a `build`.
- Esta función es llamada exactamente una vez.

La función `build` es definida de la siguiente manera:

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

- b: un arreglo de tamaño $n \times n$, donde $b[i][j] = 1$ si hay un puente conectando la torre i y la torre j, o de lo contrario $b[i][j] = 0$.
- Nota que el arreglo cumple que $b[i][j] = b[j][i]$ 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

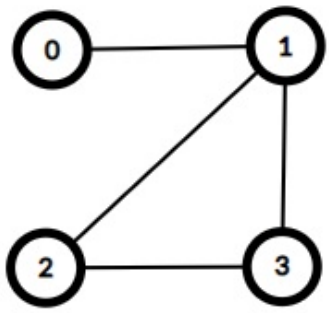
Considera la siguiente llamada:

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

Esto significa que sólo puede existir un camino de la torre 0 hacia la torre 1. Para todos los otros pares de torres (x, y), tales que $0 \leq x < y \leq 3$, deben existir exactamente dos caminos de la torre x a la torre y. Esto puede ser logrado con 4 puentes, conectando los pares de torres $(0, 1)$, $(1, 2)$, $(1, 3)$ y $(2, 3)$.

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

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



Después, debe regresar 1.

En este caso, hay varias construcciones que cumplen los requerimientos, cualquiera será considerada correcta.

Ejemplo 2

Considera la siguiente llamada:

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

Significa que no debe existir camino entre las dos torres. Esto sólo se puede cumplir si no existen puentes.

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

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

Después, la función `construct` debe regresar 1.

Ejemplo 3

Considera la siguiente llamada:

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

Esto significa que deben existir exactamente 3 caminos de la torre 0 a la torre 1. Es imposible satisfacer este conjunto de requerimientos. Por lo tanto, la función `construct` debe regresar 0 sin hacer ninguna llamada a `build`.

Límites

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

Subtareas

1. (11 puntos) $p[i][j] = 1$ (para toda $0 \leq i, j \leq n-1$)
2. (10 puntos) $p[i][j] = 0$ o 1 (para toda $0 \leq i, j \leq n-1$)
3. (19 puntos) $p[i][j] = 0$ o 2 (para toda $i \neq j, 0 \leq i, j \leq n-1$)
4. (35 puntos) $0 \leq p[i][j] \leq 2$ (para toda $0 \leq i, j \leq n-1$) y existe al menos una construcción que satisface los requerimientos.
5. (21 puntos) $0 \leq p[i][j] \leq 2$ (para toda $0 \leq i, j \leq n-1$)
6. (4 puntos) Sin límites 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 tiene el siguiente formato:

- línea 1: el valor que regresa `construct`.

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

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