Ponti aerei (supertrees)

Gardens by the Bay è un grande parco naturale in Singapore, contenente n torri (dette *supertrees*) numerate da 0 a $n-1$. Vogliamo costruire un insieme di **zero o più** ponti aerei che colleghino bidirezionalmente alcune coppie di torri, senza costruire due diversi ponti sulla stessa coppia di torri.

Un percorso dalla torre x alla torre y è una sequenza di torri tale per cui:

- il primo elemento è x e l'ultimo elemento è y;
- ogni due elementi consecutivi sono connessi da un ponte;
- tutti gli elementi sono **distinti**.

Nota che c'è sempre esattamente un percorso da una torre a sé stessa, e il numero di percorsi distinti dalla torre i alla torre j è lo stesso del numero di percorsi distinti dalla torre j alla torre i.

L'architetto capo deve costruire i ponti di modo che per ogni $0 \leq i, j \leq n-1$ ci siano esattamente $p[i][j]$ percorsi distinti tra le torri i e j, dove $0 \leq p[i][j] \leq 3$. Costruisci un insieme di ponti che rispetti questi requisiti, o determina che ciò è impossibile.

Note di implementazione

Devi implementare la seguente funzione:

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

- p: un array $n \times n$ contenente i requisiti dell'architetto.
- Se la costruzione è possibile, questa funzione deve fare esattamente una chiamata a `build` (vedi sotto) per riportare la soluzione, e poi restituire 1.
- Altrimenti, la funzione deve restituire 0 senza fare alcuna chiamata a `build`.
- Questa funzione viene chiamata esattamente una volta.

La funzione `build` è definita come segue:

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

- b: un array $n \times n$, dove $b[i][j] = 1$ se c'è un ponte che connette le torri i e j, $b[i][j] = 0$ altrimenti.
- L'array deve rispettare i requisiti dell'architetto e soddisfare $b[i][j] = b[j][i]$ e $b[i][i] = 0$ per ogni $0 \leq i, j \leq n-1$.

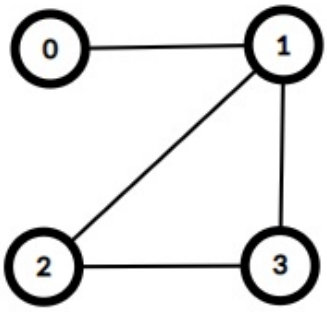
Esempi

Esempio 1

Considera la seguente chiamata:

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

Questo significa che deve esserci esattamente un percorso tra le torri 0 e 1, e due percorsi per ogni altra coppia di torri distinte. Questi requisiti possono essere soddisfatti con 4 ponti, connettendo le coppie di torri $(0, 1)$, $(1, 2)$, $(1, 3)$ e $(2, 3)$.



Per riportare questa soluzione, la funzione `construct` deve effettuare la seguente chiamata:

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

e poi restituire 1.

In questo caso ci sono anche altre costruzioni che rispettano i requisiti, ognuna delle quali verrà considerata corretta.

Esempio 2

Considera la seguente chiamata:

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

Questo significa che non deve esserci modo di spostarsi tra le torri, il che può essere ottenuto solo non avendo alcun ponte. Quindi, la funzione `construct` deve fare la seguente chiamata:

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

e poi restituire 1.

Esempio 3

Considera la seguente chiamata:

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

Questo significa che devono esserci esattamente 3 percorsi dalla torre 0 alla torre 1, il che non può accadere. Quindi, la funzione `construct` deve restituire 0 senza fare alcuna chiamata a `build`.

Assunzioni

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

Subtask

1. (11 punti) $p[i][j] = 1$ (per ogni $0 \leq i, j \leq n - 1$)
2. (10 punti) $p[i][j] = 0$ o 1 (per ogni $0 \leq i, j \leq n - 1$)
3. (19 punti) $p[i][j] = 0$ o 2 (per ogni $i \neq j$, $0 \leq i, j \leq n - 1$)
4. (35 punti) $0 \leq p[i][j] \leq 2$ (per ogni $0 \leq i, j \leq n - 1$) ed esiste almeno una costruzione che soddisfa i requisiti.
5. (21 punti) $0 \leq p[i][j] \leq 2$ (per ogni $0 \leq i, j \leq n - 1$)
6. (4 punti) Nessuna limitazione aggiuntiva.

Grader di esempio

Il grader di esempio legge l'input nel seguente formato:

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

Il grader di esempio scrive l'output nel seguente formato:

- riga 1: il valore restituito da `construct`.

Se il valore restituito da `construct` è 1, il grader d'esempio stampa anche:

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