



# Cancha de Fútbol

El bosque de Nagyerdő en la ciudad de Debrecen es una cuadrícula de  $N \times N$  celdas. Las filas de la cuadrícula se numeran de 0 a  $N - 1$  de norte a sur, mientras que las columnas de se numeran de 0 a  $N - 1$  de oeste a este. La celda  $(r, c)$  denota la celda en la fila  $r$  y la columna  $c$ .

Cada celda del bosque o bien está **vacía** o bien contiene un **árbol**. Al menos una celda del bosque está vacía.

DVSC, el famoso club de la ciudad, quiere construir una cancha de fútbol en el bosque. Una cancha de tamaño  $s$  (con  $s \geq 1$ ) es un conjunto de  $s$  celdas *distintas y vacías*  $(r_0, c_0), \dots, (r_{s-1}, c_{s-1})$ . Formalmente, esto quiere decir que:

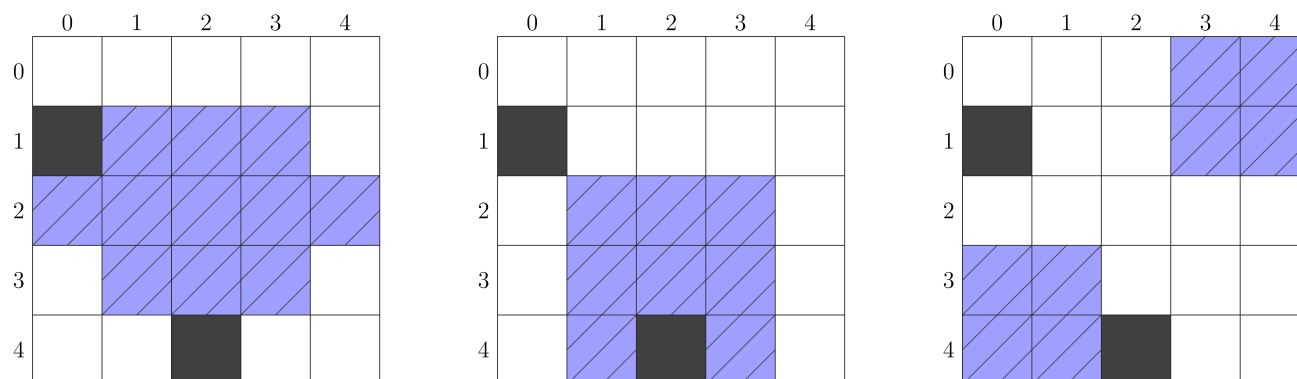
- para cada  $i$  de 0 a  $s - 1$ , inclusive, la celda  $(r_i, c_i)$  está vacía,
- para cada  $i, j$  tal que  $0 \leq i < j < s$  o bien se cumple que  $r_i \neq r_j$ , o se cumple  $c_i \neq c_j$ , o se cumplen ambas desigualdades.

El fútbol se juega con una pelota que se va moviendo por las celdas de la cancha. Una **patada recta** es un movimiento de la pelota que consiste en cualquiera de estas dos posibles acciones:

- Mover la pelota desde la celda  $(r, a)$  hasta la celda  $(r, b)$  ( $0 \leq r, a, b < N, a \neq b$ ), donde la cancha contiene *todas* las celdas entre las celdas  $(r, a)$  y  $(r, b)$  en la fila  $r$ . Más formalmente,
  - si  $a < b$  entonces la cancha tiene que contener la celda  $(r, k)$  para cada  $k$  tal que  $a \leq k \leq b$ ,
  - si  $a > b$  entonces la cancha tiene que contener la celda  $(r, k)$  para cada  $k$  tal que  $b \leq k \leq a$ .
- Mover la pelota desde la celda  $(a, c)$  hasta la celda  $(b, c)$  ( $0 \leq c, a, b < N, a \neq b$ ), donde la cancha contiene *todas* las celdas entre las celdas  $(a, c)$  y  $(b, c)$  en la columna  $c$ . Más formalmente,
  - si  $a < b$  entonces la cancha tiene que contener la celda  $(k, c)$  para cada  $k$  tal que  $a \leq k \leq b$ ,
  - if  $a > b$  entonces la cancha tiene que contener la celda  $(k, c)$  para cada  $k$  tal que  $b \leq k \leq a$ .

Una cancha es **regular** si es posible mover la pelota desde cualquier celda de la cancha a cualquier otra celda de la cancha en a lo sumo 2 patadas rectas. Notar que toda cancha de tamaño 1 es regular.

Por ejemplo, considerá el bosque de tamaño  $N = 5$ , con árboles en las celdas  $(1,0)$  y  $(4,2)$  y todas las demás celdas vacías. La figura de abajo muestra tres canchas posibles en este bosque. Las celdas con árboles se dibujan en color oscuro y las celdas de la cancha se dibujan a rayas.



La cancha de la izquierda es regular. Sin embargo, la cancha del medio no es regular, porque se necesitan al menos 3 patadas rectas para mover la pelota desde la celda  $(4,1)$  hasta la celda  $(4,3)$ . La cancha de la derecha tampoco es regular, porque es imposible mover la pelota desde la celda  $(3,0)$  hasta la celda  $(1,3)$  con patadas rectas.

El club quiere construir la cancha regular de mayor tamaño posible. Tienes que encontrar el mayor  $s$  tal que existe una cancha regular de tamaño  $s$  en el bosque dado.

## Detalles de Implementación

Tienes que implementar la siguiente función:

```
int biggest_stadium(int N, int[][] F)
```

- $N$ : el tamaño del bosque.
- $F$ : un arreglo de tamaño  $N$  que contiene arreglos de tamaño  $N$ , que forman una descripción del bosque. Para cada  $r$  y  $c$  con  $0 \leq r < N$  y  $0 \leq c < N$ , si  $F[r][c] = 0$  entonces la celda  $(r, c)$  está vacía, y si  $F[r][c] = 1$  entonces la celda contiene un árbol.
- La función debe devolver el mayor tamaño de una cancha regular que se puede construir en el bosque dado.
- La función se llamará exactamente una vez por cada caso de prueba.

## Ejemplo

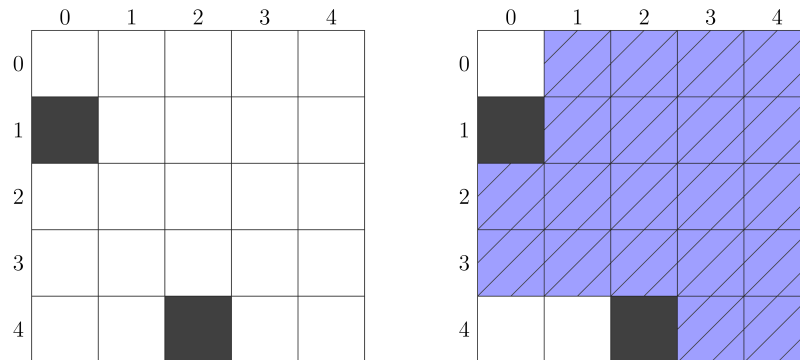
Considerá la siguiente llamada:

```

biggest_stadium(5, [[0, 0, 0, 0, 0],
                    [1, 0, 0, 0, 0],
                    [0, 0, 0, 0, 0],
                    [0, 0, 0, 0, 0],
                    [0, 0, 1, 0, 0]])

```

El bosque dado en este ejemplo se muestra a la izquierda, y una cancha regular de tamaño 20 se muestra en la derecha en la siguiente figura:



Como no hay ninguna cancha regular de tamaño 21 o más, la función debe devolver 20.

## Restricciones

- $1 \leq N \leq 2000$
- $0 \leq F[i][j] \leq 1$  (para cada  $i$  y  $j$  tal que  $0 \leq i < N$  y  $0 \leq j < N$ )
- Hay al menos una celda vacía en el bosque. Es decir,  $F[i][j] = 0$  para algún  $0 \leq i < N$  y  $0 \leq j < N$ .

## Sub-tareas

1. (6 puntos) A lo sumo una celda contiene un árbol.
2. (8 puntos)  $N \leq 3$
3. (22 puntos)  $N \leq 7$
4. (18 puntos)  $N \leq 30$
5. (16 puntos)  $N \leq 500$
6. (30 puntos) Sin restricciones adicionales.

En cada sub-tarea, puedes obtener el 25% del puntaje de la sub-tarea si tu programa detecta correctamente si la cancha que contiene *todas* las celdas vacías del bosque es regular.

Más precisamente, para cada caso de prueba donde la cancha que contiene todas las celdas vacías del bosque sea regular, tu solución

- obtiene puntaje completo si devuelve la respuesta correcta (que será el tamaño de la cancha que contiene todas las celdas vacías del bosque).

- de lo contrario, obtiene 0 puntos.

Para cada caso de prueba donde la cancha que contiene todas las celdas vacías del bosque **no** sea regular, tu solución

- obtiene puntaje completo si devuelve la respuesta correcta.
- obtiene 0 puntos si devuelve el tamaño de la cancha que contiene todas las celdas vacías del bosque.
- obtiene 25% del puntaje si devuelve cualquier otro valor.

El puntaje obtenido en cada subtarea es el mínimo del puntaje obtenido en cada uno de los casos de prueba de la subtarea.

## Grader de Ejemplo

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

- línea 1:  $N$
- línea  $2 + i$  ( $0 \leq i < N$ ):  $F[i][0] \ F[i][1] \ \dots \ F[i][N - 1]$

El grader de ejemplo imprime la respuesta en el siguiente formato:

- línea 1: lo que devuelva la función `biggest_stadium`