Laberinto S14772_es

Sea una matriz $n \times m$ de enteros. Si un valor de la matriz es negativo, diremos que en esa posición hay una **piedra**. Si el valor es mayor o igual que cero, entonces diremos que en esa posición hay un **premio**.
Debéis hacer una función que reciba una matriz `M`, un entero `potencia > 0`, un entero `objectiu > 0` y un string `recorregut` que podrá contener únicamente los caracteres `N`, `S`, `E`, `O`. Estos valores determinan un recorrido por la matriz: partiendo de la posición inicial $(0, 0)$, si recibimos una `N` tendremos que subir una fila, si recibimos una `S` tendremos que bajar una fila, si recibimos una `O` tendremos que ir una columna a la izquierda, y si recibimos una `E` tendremos que ir una columna a la derecha. Obviamente, no podemos salir de la matriz. Si el carácter nos hace salir de la matriz, deberemos quedarnos en la misma posición. La función deberá indicar si con el `recorregut` podremos conseguir recoger suficientes premios para llegar al `objectiu` o no.

La posición de partida es $(0, 0)$. Además, esta posición no contendrá nunca ninguna piedra, solo un premio.
Cada vez que pasamos por una posición (i, j) puede ocurrir

1. Que haya una piedra (un valor estrictamente menor que cero). En este caso, disminuimos la `potencia` en tantas unidades como el valor absoluto de esta posición.

2. Que haya un premio. En este caso, acumulamos dicho valor.

Podemos pasar por la misma posición más de una vez, dependiendo de lo que nos indique el `recorregut`. Y cada vez que pasemos por ella, podremos acumular el premio. Ahora bien, si nos encontramos en una posición dada y el siguiente movimiento del `recorregut` nos hace salir de la matriz, lo que haremos será no movernos de posición, y no habrá ni acumulación de premio (si nos encontramos en un premio), ni reducción de potencia (si nos encontramos en una piedra).
La función devolverá `ACONSEGUIT` si el recorrido es capaz de recoger al menos tantos premios como `objectiu`. En cambio, si ha pasado por tantas piedras que ha agotado toda su `potencia`, deberá devolver `ESGOTAT`. Si después de agotar el recorrido no ha podido recoger suficientes premios como `objectiu`, entonces deberá devolver `NO HI ARRIBEM`.

Por ejemplo, si tenemos la matriz

1	2	-1	3
-1	1	1	-4
2	4	-1	2
3	-1	-1	2
1	0	1	1

y tenemos que `potencia = 6`, `objectiu = 10` y `recorregut = EESESSS`, comenzaremos en la posición $(0, 0)$. El recorrido será el siguiente:

Recorregut	Posición	M[i][j]	Potencia	Acumulado
E	(0,1)	2	6	3
E	(0,2)	-1	5	3
S	(1,2)	1	5	4
E	(1,3)	-4	1	4
S	(2,3)	2	1	6
S	(3,3)	2	1	8
S	(4,3)	1	1	9

El resultado será `NO HI ARRIBEM` porque habremos agotado el recorrido, pero no habremos conseguido suficientes premios como el objetivo nos pedía.
Haz una función `matriu_laberint` con la siguiente declaración y especificación:

```
/*
 * PRE:  M.size() > 0 and M[0].size() > 0, es una matriz de enteros.
 *  M[0][0] >= 0.
 *  potencia > 0, objectiu > 0.
 *  recorregut.size() > 0 y recorregut solo contiene 'N','S','E','O'.
 *
 * POST: Devuelve ACONSEGUIT si el recorrido es capaz de recoger
 *       al menos tantos premios como objectiu.
 *       Devuelve ESGOTAT si recoge tantas piedras o más que potencia.
 *       Devuelve NO HI ARRIBEM si el recorrido no consigue recoger
 *               tantos premios como objectiu.
 */

string matriu_laberint(const Matriu& M,int potencia, int objectiu,
                       const string& recorregut)
```

Observación

Solo es necesario enviar la función que se os pide (y las funciones que hayáis podido declarar vosotros).
Además, en el fichero que enviéis **debe** aparecer también esto:

```
#include <iostream>
#include <vector>
using namespace std;

typedef vector <int>   Vector;
typedef vector<Vector> Matriu;
```

No se puede usar la ordenación de C++: `std::sort`. Tampoco se puede usar el método `push_back()` de la clase `vector`.
Si queréis, podéis usar las funciones `min`, `max` o `swap`.

Entrada

La entrada es una matriz de enteros no vacía, un entero potencia, un entero objectiu y un string recorregut. En la posición M[0][0] no hay una piedra. *potencia* > 0, *objectiu* > 0. *recorregut.size*() > 0 y recorregut solo contiene 'N','S','E','O'.

Salida

Devuelve ACONSEGUIT si el recorrido es capaz de recoger al menos tantos premios como objectiu. Devuelve ESGOTAT si recoge tantas piedras o más que potencia. Devuelve NO HI ARRIBEM si el recorrido no consigue recoger tantos premios como objectiu.

Ejemplo de entrada 1

```
5 4
 1  2 -1  3
-1  1  1 -4
 2  4 -1  2
 3 -1 -1  2
 1  0  1  1

10
10
ESENOSSSSOOONNE

2
10
EESESSSOOOONN

6
10
EESESSS
```

Ejemplo de salida 1

```
POTENCIA: 10 OBJECTIU: 10 RECORREGUT: ESENOSSSSOOONNE RESULTAT: ACONSEGUIT
POTENCIA: 2 OBJECTIU: 10 RECORREGUT: EESESSSOOOONN RESULTAT: ESGOTAT
POTENCIA: 6 OBJECTIU: 10 RECORREGUT: EESESSS RESULTAT: NO HI ARRIBEM
```

Ejemplo de entrada 2

```
5 4
0  2  4  8
29 -1 4  -1
28 39 35 15
27 -7 36 -6
25 22 20 20

10
90

SEOOSSSOEESEEENNN
```

Ejemplo de salida 2

```
POTENCIA: 10 OBJECTIU: 90 RECORREGUT: SEOOSSSOEESEEENNN RESULTAT: ACONSEGUIT
```

Información del problema

Autoría: PRO1

Generación: 2026-01-25T12:42:34.223Z