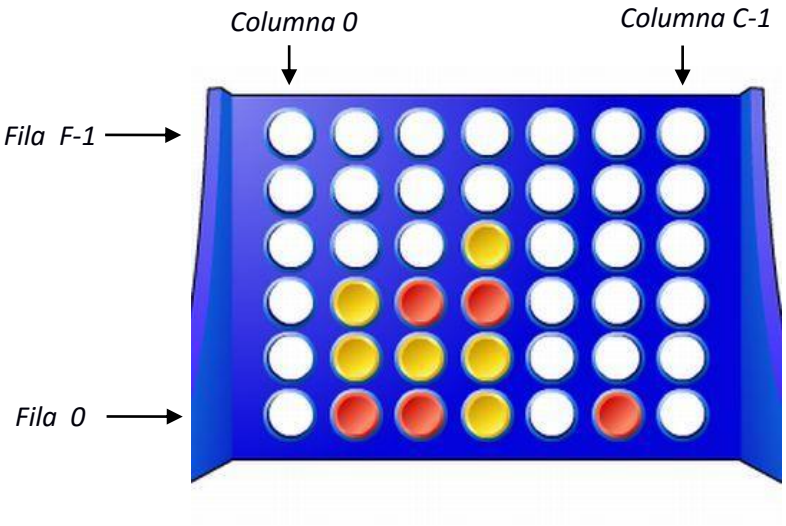
DESCRIPCIÓN DEL JUEGO CONNECTA N

Conecta 4 (También conocido como "4 en raya") es un juego en el que el objetivo es ser el primero en hacer una línea de cuatro fichas consecutivas.

En nuestro caso hemos modificado el juego pasando a ser "Conecta N", consistente en conectar N fichas consecutivas en vez de cuatro, dentro de un tablero de F filas por C columnas. El resto de las reglas se mantienen igual.



REGLAS

demo: http://mathworld.wolfram.com/images/gifs/connect4.gif

- El juego se desarrolla en un tablero de F filas por C columnas en posición vertical
- Los jugadores se turnan para echar sus fichas en las columnas (la ficha caerá por la columna hasta apoyarse en otra ficha o llegar al final, fila 0)
- Las fichas ocuparán la posición más baja de la columna cada vez
- El jugador gana cuando consigue colocar N de sus fichas en línea (horizontal, vertical o diagonal), con lo que acaba el juego

EJERCICIO

El ejercicio consiste en implementar la interfaz **com.darwinex.connectn.game.Game** para permitir evaluar los diferentes movimientos de los jugadores y determinar qué jugador ha ganado la partida (en caso de haber ganador).

Como comienzo se da la clase **GameImpl** cuyo constructor recibe como argumentos: nº de columnas, nº de filas, y nº de fichas necesarias a conectar para ganar el juego. Para realizar el ejercicio se debe completar la implementación de esta clase.

Las clases dentro del paquete **com.darwinex.connectn** son simplemente clases de apoyo necesarias para la implementación de la interfaz.

En la ruta **src/test/java** se encuentran las clases de test necesarias para validar la correcta resolución del ejercicio. Ejecutando la clase **com.darwinex.connectn.GameTest** se realizarán 10 casos de prueba que validan diferentes posibilidades de juego.

NOTA 1: Las posiciones deben ser devueltas de acuerdo a las indicaciones de fila y columna del dibujo, ya que los tests validarán las posiciones en este formato, i.e. la esquina inferior izquierda es la posición (0, 0).

NOTA 2: Únicamente es necesario implementar la clase GameImpl para la resolución del ejercicio, sin ser necesario modificar ninguna otra clase existente. No obstante, queda a elección del candidato añadir las clases auxiliares que considere necesario, e incluso modificar las ya existentes si así lo estima.

NOTA 3: Se valorará la claridad y limpieza del código así como un buen diseño y una correcta elección de estructuras de datos para llegar a una solución lo más eficiente posible tanto en complejidad temporal como espacial. Asimismo, se sugiere al candidato documentar sus elecciones y los posibles compromisos tomados si dispone del tiempo necesario para ello.