Problema 1: El juego del Tic-tac-toe (3 puntos)

Debes realizar un programa que permita jugar al juego del Tic-tac-toe.

Este juego discurre sobre un tablero de 3x3 en el que dos jugadores van dibujando "o" y "x" de forma sucesiva sobre las casillas vacías. El primero que consigue tres figuras en línea gana. Si el tablero se llena y ningún jugador ha conseguido tres en línea, la partida termina en tablas.

Funcionamiento del programa

El programa comenzará mostrando un tablero vacío y pedirá movimientos a los jugadores. El movimiento consistirá en una pareja de números separados por un espacio que indican la fila y columna en la que se dibuja la figura. Para cada movimiento introducido el programa deberá realizar lo siguiente:

1. Comprobar si el movimiento es válido: las coordenadas deben ser números entre 1 y 3 y la casilla no debe estar ocupada. Si no es válido, se volverá a pedir el movimiento al mismo jugador, indicando que ha habido un error.
2. Si el movimiento del jugador es válido, se deberá mostrar el tablero resultante y comprobar si la jugada que se acaba de hacer ha sido una jugada ganadora o si la partida ha terminado en tablas. Mostrar la información por pantalla.
3. Si la partida no ha terminado se pedirá el movimiento al otro jugador.

Un ejemplo de una ejecución

A continuación mostramos un ejemplo de una posible ejecución del programa:

```
-------------
|   |   |   |
-------------
|   |   |   |
-------------
|   |   |   |
-------------

Turno de "o".
Introduce movimiento: 2 2
Movimiento correcto.
-------------
|   |   |   |
-------------
|   | o |   |
-------------
|   |   |   |
-------------

Turno de "x".
Introduce movimiento: 1 2
Movimiento correcto.
-------------
|   | x |   |
-------------
|   | o |   |
-------------
|   |   |   |
-------------
```

```
Turno de "o".
Introduce movimiento: 1 1
Movimiento correcto.
-------------
| o | x |   |
-------------
|   | o |   |
-------------
|   |   |   |
-------------

Turno de "x".
Introduce movimiento: 3 3
Movimiento correcto.
-------------
| o | x |   |
-------------
|   | o |   |
-------------
|   |   | x |
-------------

Turno de "o".
Introduce movimiento: 3 1
Movimiento correcto.
-------------
| o | x |   |
-------------
|   | o |   |
-------------
| o |   | x |
-------------
```

```
Turno de "x".
Introduce movimiento: 1 3
Movimiento correcto.
-------------
| o | x | x |
-------------
|   | o |   |
-------------
| o |   | x |
-------------

Turno de "o".
Introduce movimiento: 2 1
Movimiento correcto.
Ha ganado "o".
-------------
| o | x | x |
-------------
| o | o |   |
-------------
| o |   | x |
-------------

¿Empezamos de nuevo?
s/n:
```

Problema 2: El juego de la vida (4 puntos)

Implementa una aplicación para reproducir el llamado 'Juego de la vida' de Conway.

No se trata en realidad de un juego sino de una simulación virtual de células que nacen, se reproducen y mueren. La simulación tiene lugar en un tablero toroidal bidimensional de N filas por M columnas. La primera fila y la primera columna, empezando por la esquina superior izquierda, tienen índice 1. Cada celda del tablero puede contener una célula.

La simulación comienza definiendo el tamaño del tablero e indicando en qué celdas hay, inicialmente, células vivas. Por ejemplo, si representamos con '-' las celdas vacías y con '#' las que contienen una célula viva, lo siguiente es la representación de un tablero de 4x4 con células vivas en las posiciones (2,2), (2,3), (3,3) y (3,4):

```
----
-##-
--##
----
```

El tablero es toroidal, es decir, que la fila siguiente a la última es la primera fila y que la columna siguiente a la última es la primera columna. Cada celda del tablero tiene 8 celdas vecinas, las que la rodean, incluso las que están en los bordes del tablero (por ser éste toroidal).

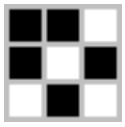
La vida en el tablero evoluciona continuamente en función de unas determinadas reglas. En cada generación, dependiendo de su localización y su vecindario, las células nacen, sobreviven o mueren. Por ejemplo, éstas son las reglas clásicas del Juego de la Vida, donde las células sólo nacen y sobreviven en un entorno ni demasiado aislado ni demasiado superpoblado:

- Una célula viva con dos o tres células vecinas vivas seguirá viviendo en la siguiente generación, en otro caso morirá (la celda que ocupaba quedará vacía).
- Una célula nacerá en una celda vacía en la próxima generación si la celda tiene exactamente tres células vecinas vivas, si no, la celda seguirá estando vacía.

Variando el número de células vecinas en cada regla se pueden obtener comportamientos muy diferentes del juego.

Lo curioso del juego de la vida es que, con reglas tan sencillas como éstas, se pueden desarrollar 'organismos' complejos que sobreviven generación tras generación. Representando una célula viva como un cuadrado negro, estas configuraciones son estáticas, es decir, permanecen inalteradas generación tras generación.

Bloque



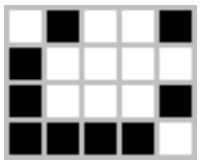
Barco

Ésta, sin embargo, es oscilante, cambiando de forma pero no de lugar.

Parpadeador

Por último, estas dos son móviles y van desplazándose por el tablero. A éstas también se les llama ‘naves espaciales’.

Planeador



Nave ligera

Funcionamiento del programa

Tu aplicación debe simular el Juego de la vida. Para ello debe solicitar, en primer lugar, el tamaño del tablero, que serán dos números mayores que 0 y menores o iguales a 30. A continuación, debe solicitar las coordenadas de las celdas donde inicialmente hay células vivas. Las coordenadas se introducen mediante dos enteros mayores que cero separados por un espacio. La aplicación deberá detectar valores fuera de rango, en función del tamaño del tablero. Las coordenadas “0 0” indican que ya no se van a introducir más células vivas. A continuación se muestra el tablero con la configuración inicial, utilizando el carácter ‘-’ (guión corto) para celdas vacías y ‘#’ para celdas ocupadas. Por ejemplo, para un tablero de 4x4:

```
Célula viva en: 2 2
OK
Célula viva en: 2 3
OK
Célula viva en: 5 3
Coordenada errónea
Célula viva en: 3 3
OK
Célula viva en: 3 4
OK
Célula viva en: 0 0

----
-##-
--##
----
```

A continuación se deben solicitar las condiciones bajo las cuales una célula viva seguiría viviendo o nacerá. Para ello se debe solicitar el número de celdas ocupadas en el vecindario para cada caso. Ten en cuenta que pueden especificarse múltiples números, con lo cual el usuario puede introducir uno o más números enteros entre 0 y 8 separados por espacios (si se repiten números, simplemente se ignora el número repetido). En el siguiente ejemplo, el usuario introduce las condiciones de las reglas clásicas del juego:

```
Número de células vecinas para sobrevivir: 2 3
Número de células vecinas para nacer: 3
```

A continuación, el programa solicitará el número de generaciones a avanzar. El usuario introducirá un número entero entre 0 y 1000. Si introduce 0 se imprime el estado actual del tablero y el programa termina. Si introduce un número entre 1 y 1000, el programa simulará la evolución del juego durante el número de generaciones indicado y mostrará el tablero resultante junto al número de generación actual. Por ejemplo, con la configuración inicial del ejemplo y las reglas clásicas, se muestra el estado del tablero en las tres primeras generaciones:

```
generación 1:
----
-###
-###
----

generación 2:
--#-
-#-#
-#-#
--#-

generación 3:
-###
-#-#
-#-#
-###
```

Problema 3: Juego automático de Tic-tac-toe (3 puntos)

Añade en el juego de Tic-tac-toe la opción de jugar contra el ordenador. Debes programar un jugador inteligente (lo llamaremos Joshua) que realice un análisis del tablero y escoja la mejor jugada posible, mostrándola por pantalla. Nosotros comenzaremos siempre la partida con "o" y Joshua jugará con "x".

Intenta que el algoritmo explore el mayor número de opciones posibles. No sólo debe considerar los movimientos ganadores, sino que debe contemplar también las posibles respuestas del oponente y escoger el mejor movimiento para todas esas posibles respuestas.

Por ejemplo, Joshua debería siempre evitar que el oponente hiciera tres en línea, como se muestra en los movimientos 1,2 y 3,1 del siguiente ejemplo.

```
Escoge una opción:
1-Juego entre humanos
2-Juego contra ordenador
Opción: 2
-------------
|   |   |   |
-------------
|   |   |   |
-------------
|   |   |   |
-------------

Hola, soy Joshua.
Introduce movimiento: 2 2
Movimiento correcto.
-------------
|   |   |   |
-------------
|   | o |   |
-------------
|   |   |   |
-------------

Mi movimiento es: 1 1
-------------
| x |   |   |
-------------
|   | o |   |
-------------
|   |   |   |
-------------

Introduce movimiento: 1 2
Movimiento correcto.
-------------
| x | o |   |
-------------
|   | o |   |
-------------
|   |   |   |
-------------

Mi movimiento es: 3 2
-------------
| x | o |   |
-------------
|   | o |   |
-------------
|   | x |   |
-------------

Introduce movimiento: 1 3
Movimiento correcto.
-------------
| x | o | o |
-------------
|   | o |   |
-------------
|   | x |   |
-------------

Mi movimiento es: 3 1
-------------
| x | o | o |
-------------
|   | o |   |
-------------
| x | x |   |
-------------

Introduce movimiento: 3 3
-------------
| x | o | o |
-------------
|   | o |   |
-------------
| x | x | o |
-------------

Mi movimiento es: 2 1
He ganado.
-------------
| x | o | o |
-------------
| x | o |   |
-------------
| x | x | o |
-------------

Lo he pasado muy bien.
Gracias por jugar.
¿Empezamos de nuevo?
s/n:
```

Observaciones:

- Se disponen de 3 horas para realizar los ejercicios.
- Los resultados se harán públicos en la página web de la Escuela Politécnica Superior de la Universidad de Alicante antes del día 17 de marzo.