SISTEMAS INTELIGENTES
Departamento de Ciencia de la Computación e Inteligencia Artificial
Universidad de Alicante **Curso 2016/2017**

Práctica 1. Búsqueda en juegos

Objetivos:

- Comprender el funcionamiento del algoritmo minimax para búsqueda en juegos.
- Desarrollar una función de evaluación para un juego en concreto.
- Manejar el entorno NetBeans para editar, compilar, depurar y ejecutar un programa.

Sesión 1. Introducción y entorno de trabajo. Diseño de la función de evaluación.

En esta primera práctica de la asignatura se empezará con el diseño de una función de evaluación adecuada para el juego Conecta-4. Con este objetivo se debe estudiar el modo de funcionamiento del juego y por qué interesa colocar la ficha en una determinada posición (por ejemplo para evitar que el jugador contrario tenga cuatro fichas en línea).

1.1 Netbeans y Java

Para el desarrollo de la práctica se utilizará NetBeans y el lenguaje Java.

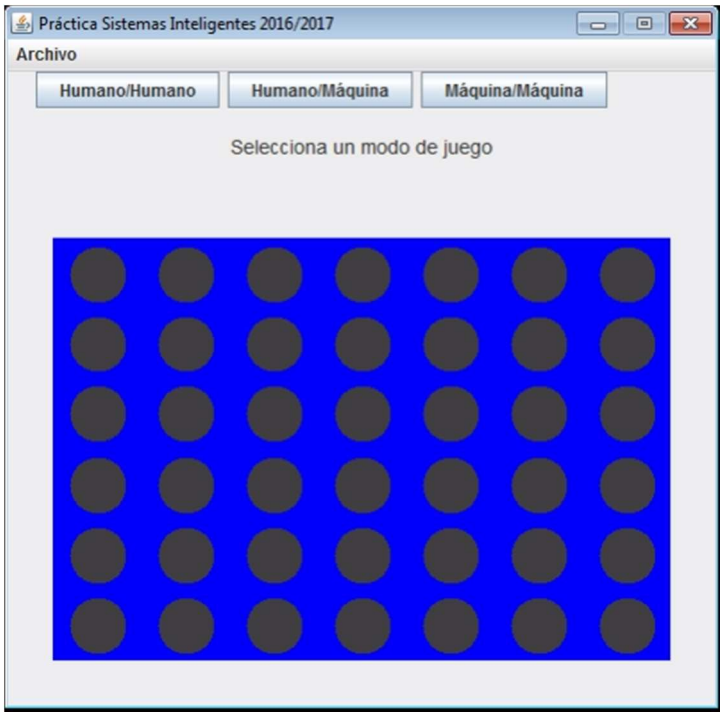
NetBeans es un entorno de desarrollo integrado, hecho principalmente para el lenguaje de programación Java, aunque extendido a otros lenguajes. NetBeans es un proyecto de código abierto, por lo tanto, lo podemos descargar libremente desde su página oficial:

http://netbeans.org/

1.2 Enunciado

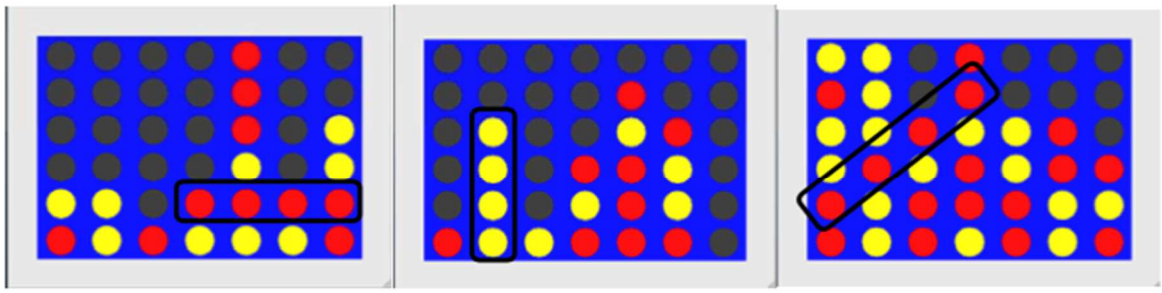
Se debe implementar el algoritmo minimax de búsqueda en juegos y una función de evaluación adecuada al juego. En concreto, se debe implementar el algoritmo para determinar la mejor jugada en el juego Conecta-4.

Se proporciona una aplicación desarrollada en Java que deberá ser completada. Esta aplicación tiene el siguiente aspecto:



Al ejecutar la aplicación se debe seleccionar una modalidad de juego para empezar a jugar. Existen tres modalidades de juego: para dos jugadores (humano contra humano), para un jugador (humano contra máquina) y ningún jugador, donde jugarían dos jugadores máquina.

Cada uno de los jugadores colocará fichas de un color. El objetivo del juego es conectar cuatro fichas del mismo color. Esta conexión puede ser horizontal, vertical o diagonal.



Sesión 2. Diseño del algoritmo minimax.

El algoritmo minimax genera todos los nodos del árbol de búsqueda hasta la profundidad deseada, evalúa cada nodo hoja y asigna un valor a la raíz dependiendo de si es max o min. El algoritmo calcula el valor V(N) de un determinado nodo.

```
algoritmo minimax. V(N)
Entrada: nodo N
Salida: valor de dicho nodo

Si N es nodo terminal entonces devolver f(N)
```

```
sino
      generar todos los sucesores de N: N1, N2, …, Nb
      Si N es MAX entonces devolver max(V(N1), V(N2), …, V(Nb)) fsi
      Si N es MIN entonces devolver min(V(N1), V(N2), …, V(Nb)) fsi
fsi
falgoritmo
```

Diagrama de clases UML

El código que se proporciona está compuesto de seis clases. La única clase que se debe modificar es JugadorMáquina. En esta clase se debe implementar el algoritmo minimax para calcular la mejor jugada. Para esto, en esta clase se dispone de una copia del tablero y una variable que indica como qué jugador se está jugando.

Sesiones 3, 4 y 5. Implementación del algoritmo minimax.

Estas sesiones se dedicarán a la implementación del algoritmo minimax. Es importante tener en cuenta que para la parte obligatoria de la práctica no se puede modificar ninguna clase del entorno a excepción de la clase jugadorMaquina.java.

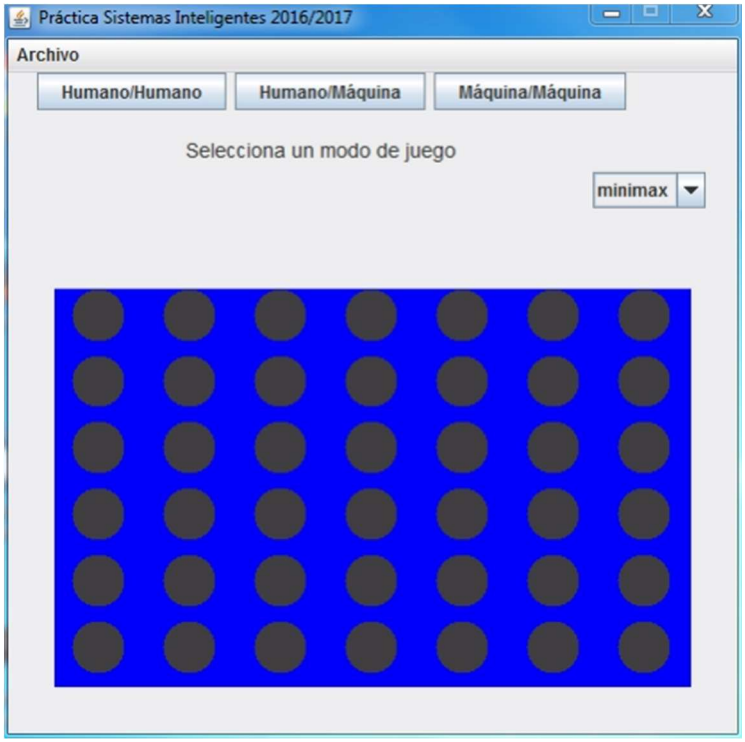
Sesión 6. Pruebas del algoritmo. Elaboración de la documentación.

La documentación es una parte muy importante de la práctica. Debe incluir la explicación detallada de los algoritmos implementados así como una sección de experimentación en la que se describan las diferentes pruebas realizadas. Se debe dejar claro el objetivo de las pruebas, qué información se ha recopilado a partir de las mismas y qué conclusiones se han obtenido. En esta sección se deben mostrar capturas de pantalla que muestren los experimentos realizados.

Sesión 7. Parte optativa. Implementación del algoritmo alfa-beta

La parte optativa consiste en la implementación del algoritmo alfa-beta. Se debe incluir la documentación correspondiente así como un estudio comparativo entre ambos algoritmos indicando número de nodos generados así como cualquier otro estudio que se considere interesante.

Para realizar esta parte es necesario modificar el entorno gráfico de la práctica incorporando algún elemento que permita elegir el algoritmo de la máquina, por ejemplo como en la siguiente figura:



Formato de entrega

La fecha límite de entrega es el 30 de octubre hasta las 12:00 de la noche. La entrega se realizará a través de Moodle. La entrega constará de un fichero .ZIP que contendrá:

- El proyecto Netbeans.

- **Documentación en formato PDF**.

¡¡¡IMPORTANTE!!! No cumplir cualquiera de las normas de formato/entrega anteriores puede suponer un suspenso en la práctica. Recordad que las prácticas son individuales y NO se pueden hacer en parejas o grupos. Cualquier código copiado supondrá un suspenso de la práctica para todas las personas implicadas en la copia.

Detalles de implementación

El código de vuestro algoritmo minimax debe incluirse en el archivo jugadorMaquina.java. Dentro de dicha clase se debe implementar el método minimax(), al que se llamará cada vez que le toque a un jugador colocar una ficha. Este método deberá indicar la columna en la que se va a colocar la ficha en la variable **m_columna**.

A tener en cuenta:

- NO se pueden incluir ficheros adicionales.
- NO se debe modificar ninguna clase del entorno a excepción de jugadorMaquina.java para la parte obligatoria