INTELIGENCIA ARTIFICIAL PARA JUEGOS
CURSO 2018/2019

LENGUAJES Y CIENCIAS DE LA COMPUTACIÓN
UNIVERSIDAD DE MÁLAGA

PRÁCTICA PUNTUABLE 3

La práctica consta de tres partes independientes. Los resultados deben entregarse a través de la tarea habilitada en el campus virtual. La evaluación de la práctica debe hacerse en clase antes de la entrega y puede realizarse por partes.

A) **PRIMERA PARTE:** TORNEO ENTRE JUGADORES

Definir un método en la clase `Juego` del paquete `mundoadversario` que reciba un número entero *n*. El método realizará *n* partidas entre los jugadores y al terminar mostrará por pantalla los porcentajes de partidas ganadas por el primer jugador, el porcentaje de empates, y el porcentaje de partidas ganadas por el segundo jugador.

Definir igualmente un método que muestre el resultado de enfrentar en 1000 partidas de Conecta-3 en un tablero de 3×3 a:

- dos jugadores de la clase `JugadorAleatorio`,
- un `JugadorAleatorio` contra un `JugadorEvaluar`,
- un `JugadorAleatorio` contra un `JugadorAlfaBeta` con limites de profundidad entre 2 y 6.

Los resultados de todas la pruebas anteriores se deberán mostrar por pantalla al final la ejecución del método correctamente tabulados. Por ejemplo:

```
                              Gana 1    Empate    Gana 2
-----------------------------------------------------------
Aleatorio vs Aleatorio         ...
Aleatorio vs Evaluar           ...
Aleatorio vs AlfaBeta (p=2)    ...
Aleatorio vs AlfaBeta (p=3)    ...
Aleatorio vs AlfaBeta (p=4)    ...
Aleatorio vs AlfaBeta (p=5)    ...
Aleatorio vs AlfaBeta (p=6)    ...
```

B) **SEGUNA PARTE:** ORDENACIÓN DE SUCESORES

Modificar las clases `JugadorMinimax` y `JugadorAlfaBeta` para que calculen el número de nodos hoja examinados al evaluar un movimiento.

Escribir un método que muestre por pantalla el número de nodos hoja examinados por Minimax y AlfaBeta para la posición inicial del Conecta-4 en un tablero 6×7 a profundidades desde 2 hasta 6. En el caso de AlfaBeta consideraremos dos ordenaciones distintas de los sucesores:

a) Usando una ordenación aleatoria.
b) Probando a insertar fichas en las columnas por orden creciente de su distancia al centro.

Los resultados de todas la pruebas anteriores se deberán mostrar por pantalla al final la ejecución del método correctamente tabulados. Por ejemplo:

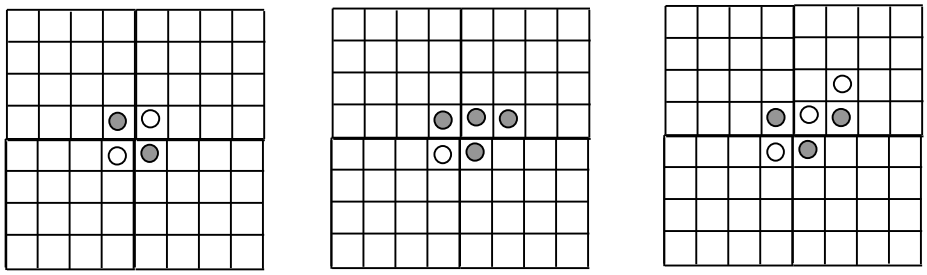
```
Profundidad          Minimax    AB(aleatorio)    AB(ordenado)
-------------------------------------------------------------------
2                            ...
3                            ...
4                            ...
5                            ...
6                            ...
```

C) **TERCERA PARTE:** EL JUEGO DEL REVERSI-N

Implementar las clases y métodos necesarios para definir el espacio de estados del juego del Reversi-N descrito a continuación.

El Reversi-N se juega en un tablero de N × N, (siendo N par) que inicialmente contiene 4 fichas en el centro. Se trata de una variante del juego del Reversi (cuya variante comercial se conoce como Othello) que se juega en un tablero de 8 × 8. Dos jugadores alternan sus movimientos. Cada jugador posee un color de ficha diferente. Las reglas son las siguientes:

- Las fichas pueden colocarse sobre el tablero pero no moverse. Cada nueva ficha debe colocarse obligatoriamente de modo que "encierre" una o más fichas del oponente, es decir, debe haber al menos una línea (horizontal, diagonal, o vertical) que vaya desde la ficha recién colocada, a través de una o más fichas del oponente, hasta otra ficha del mismo jugador sin pasar por ninguna casilla vacía. Las fichas del oponente se ven sustituidas entonces por otras tantas del jugador. Si se "encierran" varias fichas del oponente en un mismo movimiento (por ejemplo, en varias direcciones), se sustituyen todas ellas.
- El juego termina cuando el tablero está lleno, o bien el jugador al que le toca mover no puede colocar ficha en ninguna posición vacía ("pasa"). Gana el que tenga más fichas de su color en el tablero.
- En la posición inicial se dispone de 4 fichas (2 de cada jugador) ya colocadas en las cuatro casillas centrales del tablero. A modo de referencia, tomaremos como posición inicial la que se muestra en la figura de la izquierda. Supondremos que el primer turno corresponde al jugador de fichas blancas. A continuación se muestra una secuencia válida de movimientos al inicio de una partida:



Tendremos en cuenta las siguientes condiciones para la implementación:

1) La solución se programará en un paquete llamado `'reversi'`, y la clase que implemente los estados se llamará '`Reversi`'.
2) El estado del juego incluirá al menos la descripción del mismo, una indicación de a qué jugador le toca mover, así como una indicación de si el último movimiento del juego fue 'pasar'.
3) Para facilitar la prueba del código se definirán los siguientes elementos en la clase `Reversi`:
   a) Un constructor con los parámetros indicados a continuación:

```
public MiniGammon(int[][] contenidoTablero,
                  boolean turno1,
                  boolean anteriorPasa)
```

   Donde `contenidoTablero` será una matriz de modo que si en una posición hay una ficha del primer jugador valdrá 1, si hay una ficha del segundo jugador valdrá -1, y si está vacía valdrá 0; `turno1` será `true` si en el estado del juego le toca jugar al primer jugador; `anteriorPasa` será `true` si el último movimiento realizado en el juego fue 'pasar' . La siguiente llamada daría lugar al estado inicial para el caso N= 8 tal como se muestra a continuación:

```
MiniGammon e = new MiniGammon({
                                {0, 0, 0, 0, 0, 0, 0, 0},
                                {0, 0, 0, 0, 0, 0, 0, 0},
                                {0, 0, 0, 0, 0, 0, 0, 0},
                                {0, 0, 0, -1, 1, 0, 0, 0},
                                {0, 0, 0, 1, -1, 0, 0, 0},
                                {0, 0, 0, 0, 0, 0, 0, 0},
                                {0, 0, 0, 0, 0, 0, 0, 0},
                                {0, 0, 0, 0, 0, 0, 0, 0}
                               },
                              true,
                              false);
```

   b) Los métodos de consulta relacionados a continuación, de modo que devuelvan la información del estado en el mismo formato que se utilizó en el constructor anterior.

```
int[][] contenidoTablero()    { ... }
```

```
boolean turno1()   { ... }
boolean anteriorPasa()  { ... }
```

Adicionalmente, se creará una clase de prueba `TestReversi` con:

1. Un método que permita visualizar por pantalla una partida entre dos jugadores aleatorios con N= 8 y la posición inicial de referencia descrita anteriormente.
2. Un método que permita visualizar por pantalla una partida entre un jugador aleatorio y otro humano N= 8 y la posición inicial de referencia descrita anteriormente.

4) En su libro "Paradigms of Artificial Intelligence Programming: Case Studies in Common Lisp" (Ed. Morgan Kaufman, 1992), Peter Norvig describe diversas funciones de evaluación para el juego del Othello. Una de ellas consiste en tomar como valor para cada posición del tablero 1 (si contiene una ficha propia), -1 (si contiene una ficha del oponente), o 0 (si no tiene ficha). A continuación se multiplica el valor de cada posición por un peso asociado, y se suman todos los resultados.

   Definir una clase de evaluadores para el juego del Reversi cuya evaluación sea una suma ponderada de los valores de cada posición del tablero tal como se ha descrito.

   Probar dicha función con los pesos proporcionados en el libro de Peter Norvig (p. 609) reproducidos a continuación:

120	-20	20	5	5	20	-20	120
-20	-40	-5	-5	-5	-5	-40	-20
20	-5	15	3	3	15	-5	20
5	-5	3	3	3	3	-5	5
5	-5	3	3	3	3	-5	5
20	-5	15	3	3	15	-5	20
-20	-40	-5	-5	-5	-5	-40	-20
120	-20	20	5	5	20	-20	120

   Juega algunas partidas contra un `JugadorEvaluar` que emplee dicha función de evaluación, e intenta encontrar una tabla de pesos mejor.

NOTAS:

- El valor de un tablero de Reversi es una función compleja de las posiciones de las piezas. El valor de cada posición depende fuertemente de otros factores como el número de piezas que hay en el tablero, y cuáles de ellas se controlan.
- El Othello es una variante del juego del Reversi que se juega en un tablero de 8×8. La mayor diferencia con la versión que hemos programado es la condición de terminación, ya que el juego puede continuar tras "pasar" un jugador. Si ninguno de los jugadores puede mover, el juego termina.
- El mejor programa jugador de Othello se llama LOGISTELLO (Buro, 2002). En 1997 derrotó de forma contundente al campeón mundial de Othello. Su función de evaluación se obtuvo aplicando aprendizaje supervisado a una función lineal con más de un millón de rasgos.
- El juego del Othello se utiliza actualmente como herramienta para la investigación en aprendizaje por refuerzo.

(Buro, 2002) Michael Buro. Improving heuristic mini-max search by supervised learning. Artificial Intelligence 134 (2002) 85--99.

(Norvig, 1992) Peter Norvig. Paradigms of Artificial Intelligence Programming: Case Studies in Common Lisp. Morgan Kaufmann.