El juego del OTHELLO[1]

El ***Othello***, también conocido por el nombre genérico *Reversi*, tiene un tablero de 8 x 8 con fichas que son negros en un lado y blancos en el otro.

El tablero inicial se puede ver en la Figura 1. Cada jugador se turna para colocar una ficha nueva de su color. Se voltea cualquiera de las fichas del oponente que se encuentran entre la nueva ficha y las otras fichas de ese color. El objetivo del juego es tener el máximo número de fichas de tu color como sea posible.

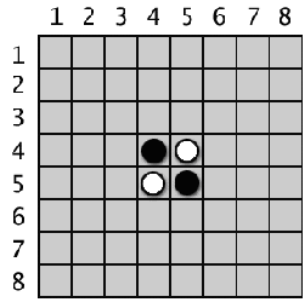

Figura 1- Tablero inicial del juego

Por ejemplo, la Figura 2 es lo que parece si el jugador blanco coloca una nueva ficha blanca en el espacio 5,6.

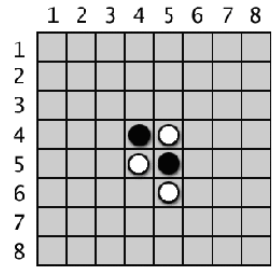

Figura 2 - Blanco coloca una nueva ficha

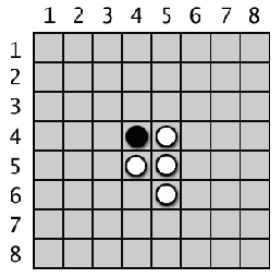

Figura 3 El movimiento del blanco volteará una de las fichas del negro

Las fichas se voltean, en todas las direcciones, siempre que se encuentren entre la nueva ficha del jugador y la ficha existente.

[1] Othello is a registered trademark of CBS Inc. Gameboard design @ 1974 CBS

OBJETIVO

Realizar un programa en lenguaje **Racket** que resuelva el problema del *Othello*, con los algoritmos :

- **Min-Max** El objetivo es elegir el mejor movimiento que debe jugar el jugador maximizador. Para ello, se consideran todos los siguientes estados posibles y se selecciona ese movimiento que da la puntuación máxima para el jugador maximizador.

  El juego alterna entre los dos jugadores (humano vs ordenador o humano vs humano) y se evalúa hasta una profundidad específica.

- **Poda Alfa-Beta** El algoritmo mantiene dos valores, alfa y beta, que representan el puntaje máximo que el jugador maximizador tiene asegurado y el puntaje mínimo que el jugador minimizador tiene asegurado respectivamente. Inicialmente alfa es un gran valor negativo y beta es un jugador de gran valor positivo. Puede suceder que al elegir una determinada rama de un determinado nodo, la puntuación mínima de la que se asegura el jugador minimizador sea menor que la puntuación máxima de la que se asegura el jugador maximizador (beta <= alfa). Si este es el caso, el nodo principal no debe elegir este nodo, ya que empeorará la puntuación para el nodo principal. Por lo tanto, las otras ramas del nodo no tienen que ser exploradas.

El trabajo se podrá elaborar por equipos de una, dos o tres personas, del mismo grupo de laboratorio. En el momento de la exposición y defensa, que se indicará con antelación, deberán estar presentes todos los miembros del equipo y uno o varios de los miembros, elegido por los profesores, tendrá que hacerse cargo de toda o parte de la defensa.

El trabajo se presentará físicamente, en un informe impreso que contenga comentada detalladamente la discusión y formulación del problema y del método de resolución empleado, así como el código Racket correspondiente bien documentado. Además, se entregará una copia del documento en formato “pdf” y el código de la aplicación en formato “rkt” en la actividad del aula virtual creada a tal efecto.

VALORACIÓN

En la solución aportada por los alumnos, se tendrán en cuenta los aspectos siguientes:

- Positivamente
  - Usar principalmente, funciones del *core* de Racket, cuando sea posible.
    - Excepciones: Librerías de interfaz gráfica y de test unitarios.
  - Uso de funciones de orden superior *map, fold, filter* y similares.
  - Implementación y uso de *funciones de orden superior y funciones lambda.*
  - Implementación con *interface* gráfico.
  - Test unitarios mediante librería *rackunit*.

- Negativamente, por ir contra la filosofía de la programación funcional.
  - Implementar algoritmos mediante bucles tradicionales.

REFERENCIAS

- Interfaz gráfica de usuario:
  - Función big-bang y estructura world, de la librería "2htdp/universe": https://docs.racket-lang.org/teachpack/2htdpuniverse.html#%28tech._world%29
  - También puede usarse librería deprecada "2htdop/world": https://docs.racket-lang.org/teachpack/world.html
  - Se explica un ejemplo de uso en el libro Realm of Racket, Capítulo 5. Ver ejemplo UFO. En siguientes capítulos hay ejemplos más elaborados.
  - El ejemplo viene instalado con DrRacket: "C:\Program Files\Racket\share\pkgs\realm\chapter5"
  - También pueden usarse otras librerías de GUI a elección del alumno.

- Algoritmia:
  - https://pats.cs.cf.ac.uk/@archive_file?p=708&n=final&f=1-Final_Report.pdf&SIG=48a08fc35e0169234716d776bec6f534c2e97babbba9edc889d4c99291178a38
  - http://web.eecs.utk.edu/~zzhang61/docs/reports/2014.04%20-%20Searching%20Algorithms%20in%20Playing%20Othello.pdf
  - https://github.com/norvig/paip-lisp/blob/master/docs/chapter18.md
  - http://ceur-ws.org/Vol-1107/paper2.pdf
  - https://pdfs.semanticscholar.org/bca7/6b49145eadbd32a07242702d2b3fe9f44a3c.pdf

ENTREGA POR BLACKBOARD:

1. Ensayo explicativo del trabajo realizado. (*pdf*)

2. Código fuente de la solución aportada. (*rkt*, con comentarios)

FECHA DE ENTREGA:

- **12 de enero de 2020 es la fecha límite a las 23:59**, para su entrega en Blackboard,

  y

- **14 de enero de 2020**, en el examen, para la entrega del documento.

Una vez recibidos los trabajos y establecido el número de grupos, se publicará una fecha con la hora de exposición para cada grupo.