T.P. 1: Reversi

Autor: Santos, Ramiro

Cátedra: Wachenchauzer – Bárbara

Fecha de entrega: 27 de Abril de 2015

1. Estructuración del programa

Para estructurar el programa en mi cabeza decidí dividirlo en cinco problemas principales:

-El desarrollo del tablero junto con su impresión.

-La validación de las coordenadas a ser ingresadas por los jugadores.

-El reemplazo de fichas por las del oponente en caso de ser posible.

-El paso de los turnos.

-La verificación de si se puede seguir jugando o no.

2. Resolución

Para armar el tablero lo he pensado como listas adentro de listas (ya que me pareció la manera más fácil siendo estas secuencias mutables), generándolas a partir de dos ciclos for como ya hemos hecho previamente en otros ejercicios para armar matrices. La lista principal tendrá N listas (en nuestro caso N es ocho puesto que se pide que el tablero sea de dimensión ocho) que conformarán las filas y cada lista tendrá en su interior 8 elementos, conformando las columnas. Luego, a la hora de imprimirlo (decidiendo tener a las letras como columnas y a los números como filas) pensé que sería mejor imprimir la primera fila con las coordenadas con letras y luego en otro ciclo for las demás filas con su número correspondiente. Después me di cuenta que era mejor imprimir todas las filas en un único ciclo for, haciendo el código más legible.

A la hora de plantearme el desarrollo de la validación de las coordenadas he de tener en cuenta que debe ser a prueba de todo para así, hacer al código irrompible (que no tire error) y más completo, de modo que cualquier cosa que se ingrese no corrompa el programa, y simplemente se pedirá que se ingrese de nuevo las coordenadas, siendo estas primero una letra (la columna) y luego un número (la fila). Ya que las restricciones al ingreso de coordenadas tienen ciertas condiciones ya me imagino que seguramente tendré que usar por lo menos más de un if.

En cuanto al reemplazo de fichas, lo que deberé hacer es, cada vez que se ingresa una coordenada, recorrer todas las direcciones posibles (8) y ver si hay fichas del oponente, y, si es que las hay, ver si la dirección recorrida finaliza con una ficha del propio jugador. En este caso reemplazaré las coordenadas ingresadas por las fichas del jugador en la lista de listas creada para generar el tablero, formándose así un nuevo tablero. Para ello consideraré a cada dirección como un vector(ahorrándome varias líneas de código): por ejemplo, si fuese a la izquierda le sumaría a las coordenadas ingresadas el vector (0,-1), siendo el 0 correspondiente a las filas ( o eje de las x) y el -1 a las columnas (o eje de las y); aquí, al ir a la izquierda me estaría manteniendo en la fila (por eso 0) y estaría retrocediendo una espacio a atrás en las columnas (por eso -1).

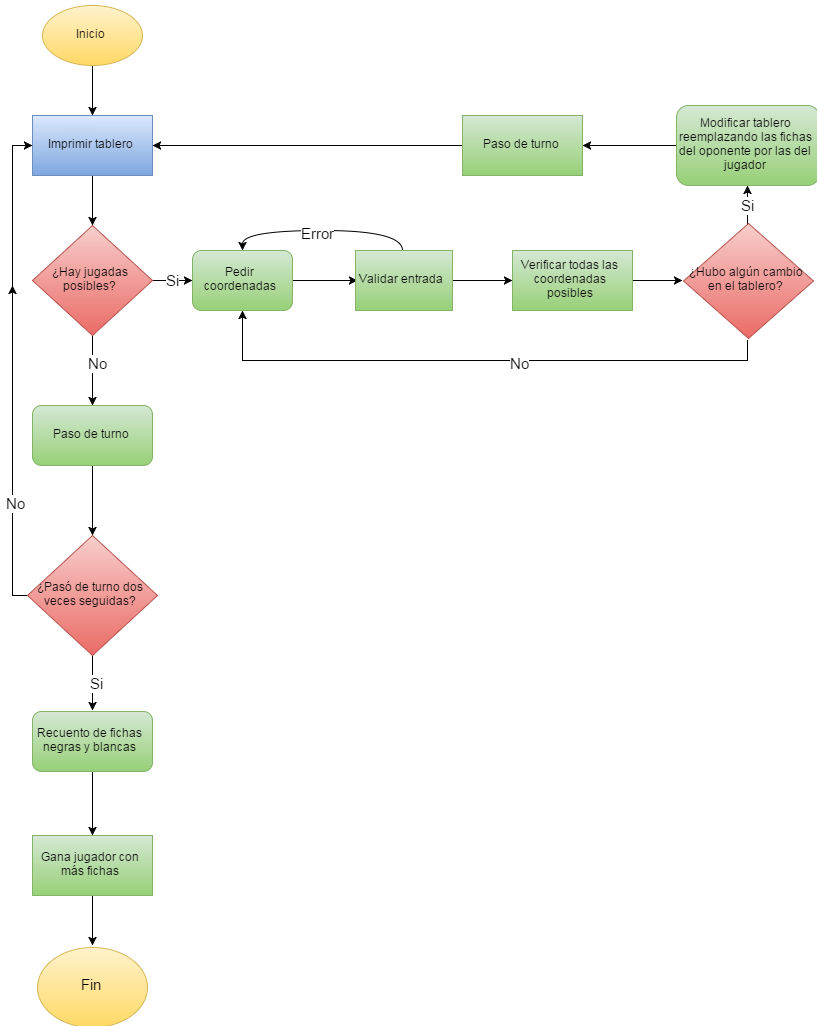
Para el paso de los turnos consideraré hacer una tupla: (‘N’, ‘B’). Sabré de quien es el turno puesto que el resto de la posición de ‘N’ es 0 (0 % 2 = 0) y porque el resto de la posición de ‘B’ es 1 (1 % 2 = 0), y así se irán intercambiando las posiciones de la tupla como los jugadores.

Finalmente, para comprobar si acabó el juego en cada turno recorreré el tablero revisando si hay alguna jugada posible para el jugador actual. En caso de ser negativo, pasará al siguiente jugador, y si tampoco puede jugar, el Reversi terminará indicando quien ganó.

Obviamente hay que tener en cuenta que todo lo dicho recién es lo fundamental. Luego quedará por hacer funciones como la determinación de quien es el ganador, la petición al usuario por si quiere jugar nuevamente, etc.

3. Flujo del programa principal

El diagrama que se puede observar a continuación representa el flujo del programa principal:



***Diagrama de flujo***