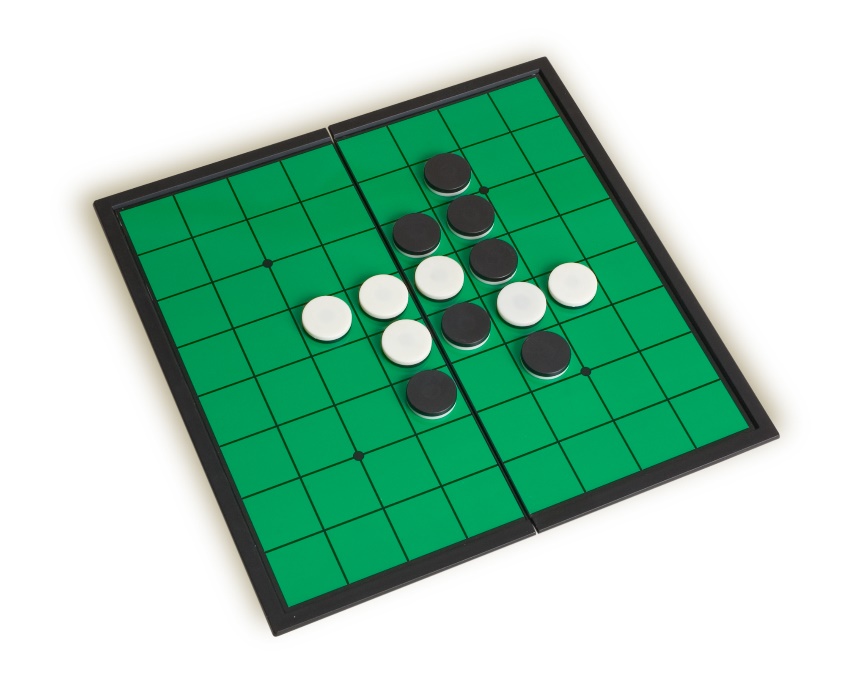
**Conclusiones proyecto PMD Infinireversi**

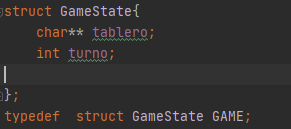
Carlos Andrés Páez Aguilar

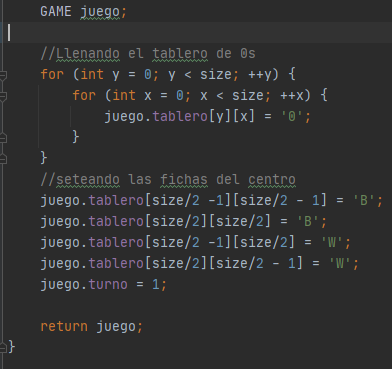
Oscar Javier Hernández Salamanca

A mediados de este semestre, la materia de PMD propuso el tan esperado proyecto final, con varias opciones para el salón, la opción por la que nosotros optamos se llama infinireversi, el juego consistía en el clásico reversi, un juego sencillo de mesa para 2 jugadores. La dificultad del juego es baja pero su implementación represento todo un reto para nosotros.



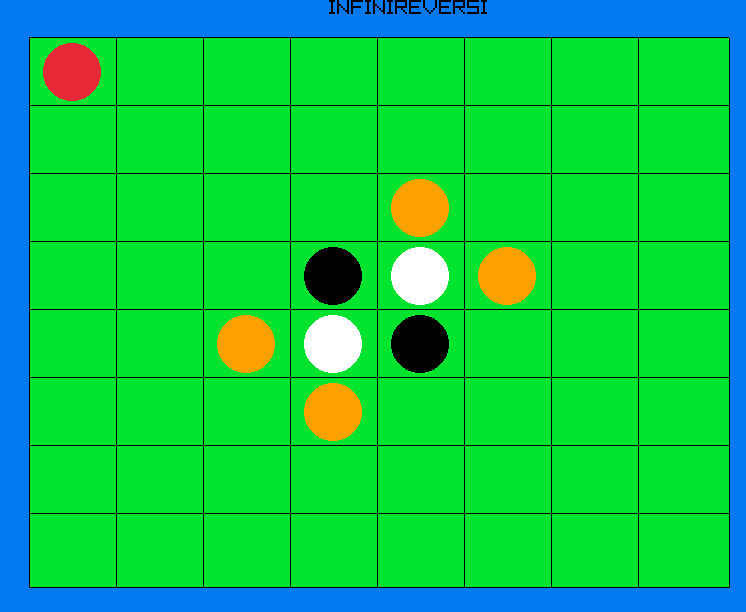
Desde un inicio pensamos “¿Qué cosas aprendidas en esta materia son las que vamos a usar para que funcione? Estamos viéndolas por una razón ¿no?”. Al inicio las únicas ideas que sabíamos eran: 1.- tendremos que separar las funciones en un .h y .c y que se trataría de una matriz.

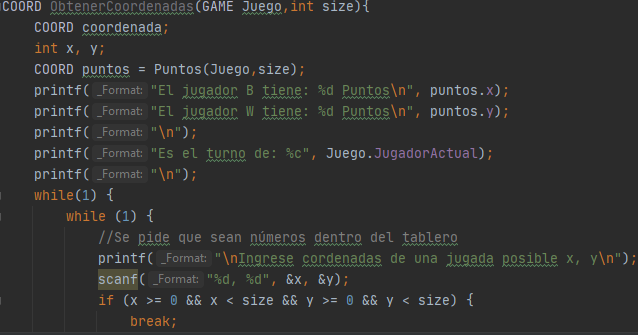
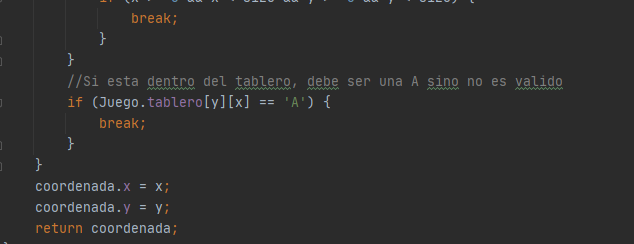
Comenzamos el desarrollo sabiendo que además del tablero hay otro dato que necesitaríamos conocer, que es el turno. Por lo que creamos una estructura, inicialmente de 8x8 principalmente para hacer las pruebas de funcionamiento.

Con esto decidimos iniciar un tablero de 8x8 y colocar las fichas iniciales en las 4 casillas centrales.

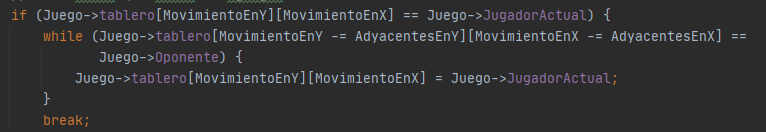
Ahora que se tenia la matriz fue cosa sencilla mostrarla en pantalla con un formato en la consola.

Los desafíos vinieron cuando había que realizar movimientos, podía ingresar una coordenada cualquiera, pero debía ser valida para que funcionara. ¿Cómo determino si es válida la casilla ingresada? La idea original era recorrer el arreglo, y encontrar las fichas “aliadas” y a partir de estas comprobar si tenían jugadas adyacentes o cercanas posibles.

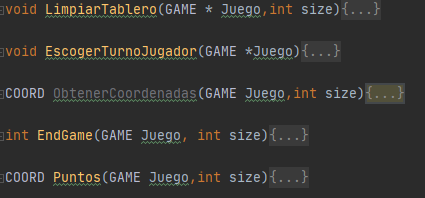
De esta manera, Terminamos viendo esto bastante más complicado de lo que esperábamos, así que en su lugar examinamos todas las casillas vacías, en búsqueda de ver si es una jugada posible para el jugador actual (busca si cumple las condiciones de encerrar a una o varias fichas opuestas entre su color).

Una vez hecho esto de distinguir casillas posibles, teníamos que realizar los movimientos, originalmente la función era demasiado compleja, incluía user input, y bastantes recorridos de ciclos para poder funcionar correctamente, el user input dentro de esta misma función, trabajaba bien en un inicio, pero conforme crecía y creía, por recomendación la segmentamos en otra parte distinta (Esto mientras usábamos la consola).

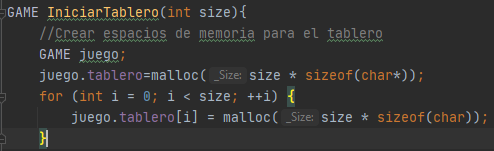
Un cambio de diseño al momento de implementar aquí fue decidir como cambiaria el color de las fichas, si detectaba la dirección y se desplazaba hasta el final o si desde el final iba en reversa convirtiendo hasta el inicio. Terminamos optando por regresar de la ficha recién puesta hasta la ficha original.



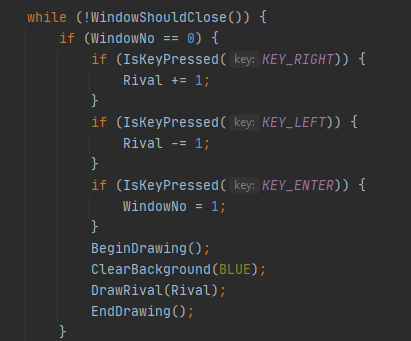
Adicional a estas funciones principales agregamos otras que nos ayudarían a limpiar el tablero antes de hacer otra jugada posible, otra que nos ayudara con el marcador, una que nos ayudase a determinar quién era el jugador en turno y por último una que nos avisara cuando el juego debía detenerse.

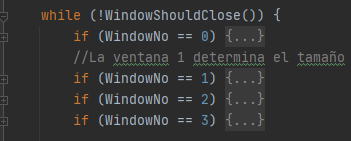


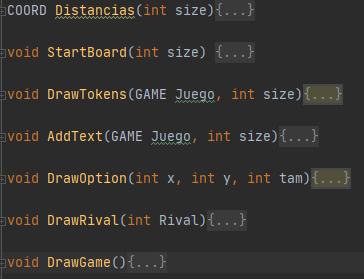
Con todo esto listo reversi ya estaba hecho, nuestro problema era que necesitábamos un infinireversi, Infinireversi implicaba un tablero de tamaño indicado por el jugador, aquí llego el uso de otro concepto que vimos en el curso, la poderosa memoria dinámica.

Fueron muchos conflictos los que hubo con la memoria dinámica sobre todo como implementarla. Teníamos problemas “¿Y si el size nos altera todo?¡ Y si no almacena bien?” todo se soluciono mucho mas sencillo de lo que parecía

Con esto listo Infinireversi estaba completado, lamentablemente solo en consola lo siguiente es el reto Raylib. Este fue todo un reto a pesar de que ambos nos encontrábamos mínimamente familiarizados con una interfaz gráfica. El mayor esfuerzo fue comprender como se comportaba la librería y de ahí lo sencillo fue adaptar partes del juego a una representación visual.

Inicialmente supimos hacer pantallas estáticas, no se actualizaban, solo se creaban y se mostraban. Debemos reconocer que fue todo un problema intentar añadir las funciones que tuvieran user input porque nos generaba un conflicto al compilar dejando sin responder la pantalla creada, terminando la ejecución o simplemente sin respuesta. Al darnos cuenta de esa complicación decidimos cambiar la forma en que el usuario metía datos al juego, paso de entrada en la consola, a manipulación con el teclado. El teclado facilito muchas cosas como el movimiento y la realización de movimientos.

Así logramos implementar Reversi en Raylib, lamentablemente infinireversi era distinto, el usuario originalmente debía meter los datos, pero no teníamos esa opción, inténtanos encapsular pantallas en funciones y que retornaran valores como el tamaño del tablero, el movimiento en el juego etc. Ahí nos dimos cuenta de que Raylib solo permite crear una pantalla por ejecución, entonces los planes de varias funciones que crearan pantallas distintas pasaron a una función que dibujara diferentes pantallas dependiendo de si una es válida o no. Al terminar de usar una pantalla había que cambiar a que pantalla nos moveríamos y por eso indicamos a que pantalla iríamos modificando un valor que llamamos Numero de pantalla, decidíamos que acciones se tomaban y que se dibujaría en pantalla.

Raylib consistió en especificar que queríamos escribir en pantalla y que teclas como entrada de datos íbamos a usar. Creamos distintas funciones para determinar distancias a las que estarían las líneas del tablero, el tamaño de las fichas, escribir un marcador, donde se dibujarían las fichas y un apoyo visual que es la posición del jugador.

El resultado final fueron 4 pantallas, una que le indica al usuario su rival, otra que especifica el tamaño y 2 que se refieren a una ia que a continuación explicaremos y otra que es el modo competitivo del juego, enfrentándote a la persona que tengas a un lado.

