

República Bolivariana de Venezuela

Universidad Simón Bolívar

Departamento de Computación y Tecnología de la Información

CI-2691 - Laboratorio de Algoritmos y Estructuras I

Trimestre Septiembre-Diciembre 2016

**Informe:** **Super Mario Solitaire Chess**

Profesor: Autores:

Josué Ramírez Manuel Rodríguez 13-11223

Jean Hernández 13-10648

**Introducción**

Inspirados en la saga de juegos Mario Bros y como asignación para el proyecto final de la asignatura laboratorio de Algoritmos y Estructuras I se realizó la implementación de un juego de ajedrez solitario con dicha temática.

Los objetivos planteados para la creación del juego son los siguientes:

* **Crear un juego amigable para el usuario:** Se propuso la creación de unas series de menús atractivos visualmente que le permitan al mismo moverse entre opciones sin ningún tipo de confusión, de manera tal que su experiencia de juego sea cómoda y fluida.
* **Visualización de tablero en 3D:** Para mejorar la experiencia de juego y nivel de realismo, se propuso como objetivo que el tablero de juego en cualquier dificultad se pudiese visualizar en 3D.
* **Reducir la cantidad de errores que puedan surgir a la hora de la ejecución del juego:** Los errores son comunes en todos los juegos por tanto un objetivo de Super Mario Solitaire Chess fue realizar un código con la menor cantidad de errores posibles, para que de esta forma el usuario pueda jugar sin ningún inconveniente.
* **Optimización de código**: Realizar el juego con una cantidad aceptable de líneas y ajustando cada función declarada para que pueda ser utilizada en varios subprogramas del mismo.

Por su parte, el alcance del proyecto es la realización de un juego de ajedrez solitario que cuente con las dificultades: Fácil, difícil, muy difícil y entrenamiento. Donde en la dificultad seleccionada el usuario pueda elegir entre cargar un tablero previamente configurado o cargar mediante el teclado su propia configuración de tablero. Además de contar con la funcionalidad de guardar una partida y de almacenar en una lista de records los usuarios con mayor cantidad de victorias según la dificultad en la que hayan jugado. Toda esta implementación tomando en cuenta los objetivos propuestos y con un tiempo estimado de realización de dos semanas.

El contenido del informe es el siguiente:

* Introducción
* Diseño de juego: Análisis aplicado a la implementación de juego, estructura de datos utilizada, subproblemas claves del programa, estado actual del proyecto.
* Conclusiones
* Bibliografía

**Diseño del juego**

***Análisis aplicado a la implementación del juego.***

La solución al problema propuesto se realizó utilizando un análisis descendente utilizando funciones para cada funcionalidad del juego. Cada menú se ejecuta desde una función única y la ejecución del programa se basa en hacer llamada a estas funciones cuando el usuario así lo desee.

Al ser el problema un juego de ajedrez fue necesaria la creación de funciones para el movimiento de cada ficha del tablero y para la validación del movimiento de las mismas, de esta manera el subprograma encargado de llevar a cabo una partida es capaz de determinar si la jugada solicitada por el usuario es válida o no, ejecutando una función dedicada a la animación de las fichas en el caso que la misma sea una jugada correcta. Además se necesitó implementar una función que valida el estado de perdida de una partida, el subprograma en cada jugada realizada es capaz de determinar si la partida fue ganada, quedan movimientos por realizar o en el peor de los casos si la partida fue perdida. También se decidió que desde el mismo subprograma se validara el tiempo restante que el usuario tiene para terminar la partida.

La implementación del tablero de juego se realizó mediante una función que dado un String de tablero (De la forma: “Ta1-b2” por ejemplo) convierte dicho string en una matriz 4x4 donde la posición de una ficha ahora se determina por su posición (Esta función se ejecuta si el string cargado pasa correctamente la verificación de una función que determina si la configuración de juego es válida o no). Luego la matriz es un parámetro de una función que se encarga de dibujar el tablero con las fichas en la posición correspondiente.

Para la opción “cargar partida” la información de cada partida es almacenada en un archivo que luego una función es capaz de leer y mostrar su información en pantalla, la implementación de la tabla de records es muy similar a esta pues la información de las victorias de cada usuario es almacenada en un archivo (según la dificultad) y se hace un llamado a una función donde se ordenan de manera descendente los usuarios con su cantidad correspondiente de victorias y en caso de haber un empate se realiza un ordenamiento alfanumérico para la visualización de la tabla en pantalla.

***Estructura de datos utilizada***

La estructura de datos utilizada fue el manejo de listas y archivos, en los archivos se almacenan todos los datos referentes a partidas guardadas y las victorias de un usuario. En las listas se lee y almacena la información de los archivos para luego ejecutar ordenamientos o impresiones en pantalla de las posiciones que sean necesarias. Además el manejo del tablero se realizó mediante la creación de una lista de listas (una matriz).

***Subproblemas claves del programa***

La lectura de datos vía teclado es un problema que se resolvió mediante la función declarada como **“Leer”** en ella se especificó una serie de eventos para que el módulo pygame fuese capaz de reconocer las teclas utilizadas por el usuario y luego de utilizar el botón “enter” retornar el valor del string introducido por el usuario.

Para la creación de los distintos menús del juego, se declaró la función **“dibujarmenu”** la cual dado un título, una lista de opciones la orientación y una ubicación en pixeles se procede a dibujar un menú en pantalla.

En cuanto a la validación de las posiciones en las que una ficha se puede mover se declararon las siguientes funciones según el movimiento de cada ficha, en cada una de estas funciones se almacena una lista de posiciones válidas para la ficha que ocupa la posición seleccionada por el usuario.

El caso de la ficha alfil la función es “**PosicionesValidasAlfil”.** Función que ejecuta un llamado a los subprogramas *“BusquedaDiagonalSimetrica”* y “*BusquedaDiagonalAsimetrica*”.

La función “*BusquedaDiagonalSimetrica*” se propone como solución para las posiciones validas en las que se puede dirigir un alfil si se desea mover aumentando/disminuyendo su posición en el eje horizontal y en el eje vertical. Por su parte “*BusquedaDiagonalAsimetrica*” se propone como solución para evaluar si los movimientos cuando se aumenta la posición en el eje horizontal pero se disminuye en el eje vertical es correcta y viceversa. También es importante señalar que la función posicionesvalidasalfil recibe un parámetro booleano para evaluar las posiciones validas de un peón, en caso de que el mismo sea verdadero se restringe la búsqueda en las diagonales señaladas para un máximo de 2 casillas en el eje horizontal.

En cuanto a la ficha torre se declaró la función **“PosicionesValidasTorre”.** Función que determina las posiciones validas de una torre haciendo la llamada a los subprogramas *“BusquedaHorizontal”* y *“BusquedaVertical”.* Como bien indica su nombre se utilizan estos subprogramas para realizar las búsquedas en el eje horizontal y el eje vertical respectivamente de una ficha.

Para la ficha caballo se declaró la función **“PosicionesValidasCaballo”** en ella se implementó un algoritmo para calcular de forma directa las posiciones en las que se puede mover un caballo.

Por último para la ficha rey se declaró la función **“PosicionesValidasRey”** la cual se encarga de realizar la búsqueda de posiciones validas restringidas a una casilla y para el caso en que la ficha sea una reina se ejecuta una búsqueda en todas las direcciones posibles exceptuando la del caballo.

Para la visualización del tablero de juego en pantalla fueron clave dos funciones, la primera es **“MatrizDeString”** la misma fue mencionada y explicada en el inciso de “Análisis aplicado a la implementación del juego.” Y la segunda es **“DibujarTablero”** la misma dado una matriz de tablero es capaz de dibujarlo en pantalla con un efecto 3D.

Para el control de una partida se declaró la función **“controlador\_juego”** en ella se validan el estado de perdida de una partida, el tiempo restante para finalizar una partida, si la jugada solicitada por el usuario es válida y se puede ejecutar, además de realizar la llamada a los subprogramas encargados de pausar una partida y deshacer una jugada según la dificultad elegida.

***Estado actual del proyecto***

El proyecto se encuentra completamente operativo y con todas sus opciones habilitadas.

**Conclusiones**

Se consiguió implementar un juego muy agradable visualmente con una serie de animaciones y estilo de juego único para impresionar al usuario. Se presentaron dificultades a la hora de ejecutar alguna búsqueda de posiciones validas pero las mismas fueron resueltas tras la aplicación de la técnica “ensayo y error”, también se presentó una dificultad a la hora de implementar el tiempo, el mismo en algunas ocasiones no se mostraba de manera correcta en el juego.

En cuanto a la realización del juego, se trató de una experiencia divertida en cuanto al aprendizaje de programación, además implica aprender a manejar el módulo pygame para el desarrollo de otros juegos. Fue muy grato para los miembros de este proyecto utilizar la temática de Mario Bros para revivir este juego de la infancia con un nuevo juego creado por nosotros.

Para futuros cursos de laboratorio de algoritmos I se recomienda buscar una forma de ayudar más al estudiante a obtener conocimientos en programación (con recursos teóricos y/o audiovisuales), muchas personas tienen su primera experiencia con lenguajes de programación en esta asignatura y no logran asimilar todo el contenido pragmático del curso de manera adecuada.

*Nota: Algunas de las imágenes/sonidos utilizados para la realización de este proyecto están sujetos a derechos de autor, los mismos fueron utilizados con fines educativos y no lucrativos, en ningún momento se propone violar los derechos de autor de Nintendo of America Inc.*

# **Bibliografía**

Desconocido. (diciembre de 2016). *Pygame*. Obtenido de Pygame wiki tutorials: http://www.pygame.org/wiki/tutorials

Desconocido. (diciembre de 2016). *Python*. Obtenido de Documentación Python: https://www.python.org/doc/

Ramirez, J. (2016). *Proyecto: Solitaire Chess.* Caracas: Universidad Simón Bolívar.