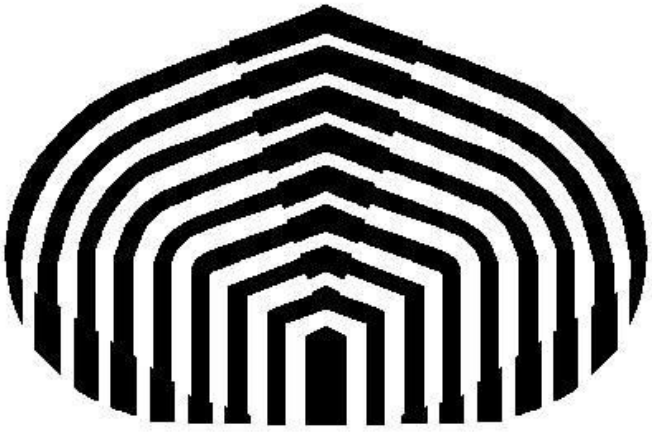


Universidad Simón Bolívar

Departamento de computación y tecnología de la información

Laboratorio de Algoritmos y Estructuras I – CI2691

Prof. Carmen Rosseline Rodríguez

Especificaciones del Proyecto

4 EN LINEA

Equipo 04

Integrantes:

Pietro Iaia. Carnet: 15-10718

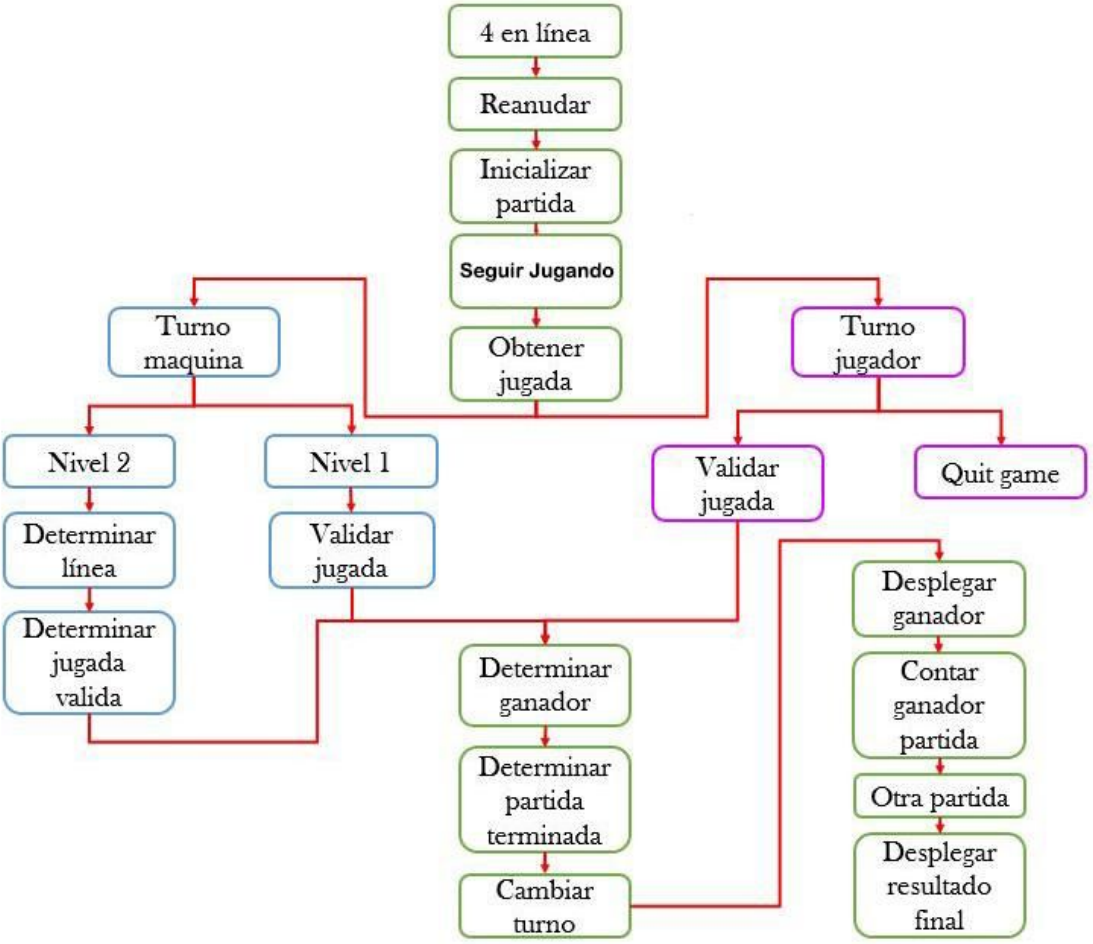
Antonella Requena. Carnet: 15-11196

Sartenejas, abril de 2018

INTRODUCCIÓN:

El juego *4 en línea* es un juego de dos jugadores que consiste en introducir fichas en un tablero vertical de dimensiones 6x7 de manera que las fichas se vayan apilando. Cada jugador tiene una ficha de un color. El objetivo del juego es en formar una línea vertical, horizontal o diagonal (en cualquier sentido) con cuatro fichas consecutivas (del mismo jugador). Si se rellenan todas las casillas y ningún jugador logró formar una línea, el juego termina en empate.

El problema a resolver consiste en lograr desarrollar en el lenguaje de programación Python una adaptación del juego 4 en línea. Para hallar la solución al problema se implementa el código utilizando la estructura del análisis descendente, basándonos en una carta estructurada (anexada a continuación). Esto permite que el problema principal se divida en tareas pequeñas que sean más sencillas de resolver.



Objetivos de los sub-programas establecidos en la carta estructurada:

- Reanudar: Este subprograma revisa si existe el archivo 'save.txt', si lo encuentra, se le dice a usuario que encontró una partida guardada y pregunta si quiere cargarla, si selecciona que si se le cargaran los resultados finales de la partida guardada. Si no se encuentra el 'save.txt' o se selecciona que no quiere cargar la partida anterior, se termina el subprograma.
- Inicializar Partida: Este subprograma se encarga de inicializar el Tablero, colocando todas las casillas en blanco e inicializar algunas variables necesarias para cada partida. También es donde se le pedirá al usuario ingresar su nombre y el nivel de dificultad que desee jugar. Si es la primera vez jugando, se le asigna el turno al computador, si no es así se le asigna el turno al ganador de la partida anterior.
- Seguir Jugando: Cada vez que sea el turno del usuario, este procedimiento le preguntará si desea seguir jugando, si responde que 'si' el programa le dejará jugar. Si responde que 'no', se le pedirá si desea guardar su partida y luego de esto el programa se cerrará.

- Obtener Jugada:
  - Turno del Jugador: El jugador elegirá la columna que desee jugar con el click del mouse y una vez escogida, se presiona la tecla 'Espacio' para lanzar la ficha a la ultima fila libre de esa columna.
    - Validar Jugada: Este es el subprograma necesario para verificar la última fila disponible en la columna deseada. Si se encuentra una, se elige esta, si no se le avisa al jugador que no puede jugar en esa columna y lo deja escoger otra.
    - Quit Game: Si el jugador desea dejar de jugar, presiona la 'x' en la esquina de arriba de la pantalla. El programa le preguntará si desea guardar la partida o no. Una vez hecha la decisión, el programa se cerrará.
  - Turno Computadora:
    - Nivel 1: En este nivel la computadora juega al azar
      - Validar Jugada: Este es el subprograma necesario para verificar la última fila disponible en la columna deseada. Si se encuentra una, se elige esta, si no se encuentra, en el caso de el turno de la computadora, se saldría del sub-programa.
    - Nivel 2: En este nivel se desarrolla una estrategia de juego
      - Determinar Línea: En este sub-programa se verifica cualquier posible jugada por encima o a los lados de la ultima ficha jugada por el computador (casillas adyacentes). Una vez verificado, revisa si puede seguir jugando por su Línea seleccionada anteriormente. Si puede escoge esa, si no escoge cualquier otra que sea Válida.
      - Determinar Jugada Valida: Este Sub-programa trabaja igual a Validar Jugada solo que enfocado al este nivel de dificultad ya que verifica no solo si tiene una ficha por debajo, también verifica donde puede estar colocada la siguiente ficha, esto ayuda a seleccionar el tipo de Línea que se quiere construir.
- Determinar Ganador: Este Sub-programa verifica cada tipo de línea en toda la matriz (Horizontal, vertical, diagonal derecha y diagonal izquierda). Si encuentra 4 fichas consecutivas del mismo color (del mismo jugador) asigna la variable Ganador al jugador que lo haya logrado. Además, la resalta cambiando las fichas de color.
- Determinar Partida Terminada: Primero verifica si hay espacios libres en el tablero, si no los hay, hace que el juego termine ya que no podría haber más jugadas. Si aún encuentra espacios libres, verifica si hay un ganador. Si lo hay, termina el juego, si no continua.
- Cambiar Turno: Cuando llegue a este Sub-programa es porque habrá terminado una corrida del ciclo de Obtener Jugada y cambia el turno al siguiente jugador.
- Desplegar Ganador: Una vez terminada la partida, se chequea el ganador de la partida y lo muestra en pantalla.

- Contar Ganador Partida: Este subprograma cuenta el número de veces que gana el Jugador, la computadora y cuantas veces queda en empate cada partida. Es un contador del resultado de la totalidad de las partidas jugadas.
- Otra Partida: Este subprograma le avisa al jugador que la partida terminó y le pregunta si quiere jugar otra. Si selecciona que sí, se reinicia el ciclo de nuevo empezando por el subprograma Inicializar Partida. Si selecciona que no, se le pregunta si quiere guardar la partida y luego sale del ciclo del juego.
- Desplegar Resultado Final: Este subprograma es llamado una vez que se vaya a terminar de jugar el juego y muestra en pantalla la cantidad de partidas ganadas por cada uno de los jugadores y las empatadas.
- Dibujar Tablero: Este subprograma dibuja el tablero, tiene las coordenadas y el color de las líneas verticales y horizontales para formar la cuadrícula 6x7.
- Dibujar Círculo: Este subprograma dibuja los círculos que representan las fichas
- Dibujar Línea: Dibuja las líneas de selección en el panel sobre el tablero cuando el usuario se mueve para escoger en qué columna jugará.
- Resaltar Línea: Cambia el color de los 4 círculos a rojo para resaltar esta línea como la ganadora.
- Bienvenida: Es un subprograma que muestra un mensaje de bienvenida al juego una vez que se inicia, muestra los controles y un mensaje de advertencia.

Variables del programa principal:

- N: int // Constante que representa las filas de la matriz (Para este programa, N sera 6)
- M: int // Constante que representa las columnas de la matriz (Para este programa, M sera 7)
- NombreJugador: str // ENTRADA: guarda el nombre del jugador
- nivel: int // ENTRADA: guarda el nivel que desea jugar el usuario
- Tablero:array[0..N)[0..M)of int // Variable usada para contener la matriz del tablero (posiciones jugadas)
- TableroG:array[0..N)[0..M)of int // Variable usada para contener llenar la matriz del tablero que quedo de la partida anterior
- turno: int // Variable del turno de los jugadores, vale 1 si el turno es del usuario y 2 si es del CPU.
- JugadaPrimeraVez: bool // Variable usada para ver si el CPU en el nivel medio ya jugo por primera vez
- primeraJugadaUsuario: bool // Variable usada para ver si el usuario ya jugo por primera vez
- Linea: int // Variable que contiene la Linea que el CPU desea jugar en el nivel medio

- jugada: int // Variable que guarda la jugada del jugador y del CPU en el nivel 1
- Ganador: int // Variable que contiene el Ganador de la partida (Siempre sera 0 hasta que se decida si se queda asi o cambia a 1 o 2)
- esValida: bool // Variable para determinar si la jugada del jugador es valida o no
- siguePartida: bool // Variable que determina si la partida sigue o ya termino (Cuando se le pregunta al usuario si desea seguir jugando y dice que si, vale True)
- jugarOtra: bool // Variable para determinar si se desea jugar otra partida o no
- p: int // Variable que contendra la ultima fila jugada por el CPU en el nivel medio
- q: int // Variable que contendra la ultima columna jugada por el CPU en el nivel medio
- Ganador0:int // Contador de veces que ha quedado en empate la partida
- Ganador1:int // Contador de veces que ha ganado el Jugador
- Ganador2:int // Contador de veces que ha ganado el computador
- PrimeraPartida: bool // Esta variable verifica si es la primera partida que se juega entre el computador y el jugador

Manera en que la máquina toma decisiones:

- Nivel Principiante o 1: Para el nivel 1 de este juego se nos pidió que la computadora escogiera una casilla válida cualquiera y eso es lo que se hizo. El computador escoge una de las 7 columnas de manera aleatoria y coloca la ficha en la última fila vacía.
- Nivel Intermedio o 2: Para el Nivel 2 de este juego se nos pidió que la máquina tratara de construir una línea de 4 fichas. La estrategia que implementamos es que el computador escogiera entre 5 tipos de líneas. Línea horizontal derecha, Línea horizontal izquierda, Línea vertical, Línea diagonal derecha o Línea diagonal izquierda. Y la escoge dependiendo de si es válido jugar en esa dirección. Si el computador ya escogió una Línea en alguna jugada anterior y es posible seguir jugando en esa dirección, el computador lo hará hasta que se encuentre con un obstáculo y ahí cambiará de Línea o, si no puede moverse en ninguna de las 5 direcciones, escoge cualquier casilla vacía válida aleatoria e intentará nuevamente construir alguna línea a partir de dicha casilla.

Conclusiones:

Durante a la realización de este proyecto aprendimos aún más sobre la sintaxis y las funcionalidades del lenguaje de programación Python, su versatilidad y dinamismo. Además alcanzamos un profundo análisis en el manejo de matrices y de cómo deben funcionar las aserciones en un programa. Todo esto se logró gracias a la modularización del código, esto es, separar las tareas o problemas a resolver en bloques de código para facilitar la legibilidad del mismo, esto es conocido como análisis descendente.

La implementación del análisis descendente también permitió una buena "comunicación" entre funciones y subprogramas que a priori responden a problemas aislados. Gracias a esto el proceso de "debugging" del código fue sencillo y poco tedioso, ya que cuando se presentaba un problema, se sabía rápidamente que sub-programa lo generaba y, al ser este aislado de los demás (es decir, todo lo que ocurría en él no afectaba otros subprogramas), se podía llegar a la raíz del problema rápido.

La ejecución de este proyecto también fue de gran ayuda para aprender el esfuerzo que conlleva desarrollar un videojuego y propició herramientas de cómo trabajar en equipo dentro de un mismo proyecto.