**ANÁLISIS DEL PROBLEMA**

1. **¿Cuál es el problema?:** Se desea realizar un programa que simule el juego “Cuatro en línea”. El objetivo de dicho juego es alinear antes que el contrincante cuatro fichas del mismo color. Las cuatro fichas pueden ser colocadas de forma horizontal, vertical o en diagonal. Para seleccionar una casilla dentro del tablero, el jugador debe seleccionar la columna sobre la cual desea colocar la ficha (siempre que tenga espacios disponibles), y ésta tomará la primera casilla libre de abajo hacia arriba, de manera que la casilla superior de cada columna será la última en ser ocupada por una ficha. permita el ingreso de dos jugadores, y que los mismos puedan realizar tantos juegos como juzguen necesario, manteniendo las estadísticas correspondientes a las victorias que han logrado cada uno de ellos, así como también de los empates ocurridos. Debe tomarse en cuenta que el inicio de cada partida será alternado entre ambos jugadores con la intención de ofrecer las mismas ventajas a cada uno de ellos.

*NOTA*: se debe utilizar un tablero de seis (6) filas y siete (7) columnas.

1. **Datos:**
   * **Datos suministrados por el problema:**
     + Número de Jugadores: 2.
     + Dimensiones del tablero: seis (6) filas por siete (columnas).
     + Número de fichas alineadas para ganar: cuatro ó más (vertical, horizontal o diagonal).
   * **Datos suministrados por el usuario:** 
     + Nombre de los jugadores (si desea modificarlos).
     + Si está jugando una partida, el jugador de turno debe teclear el número de la columna en donde desee colocar su ficha.
2. **Operaciones:**
   * **Funciones y procedimientos implementados:**
     + Procedimiento ‘estadisticas’: muestra el balance general de las partidas jugadas, ganadas, empates y perdidas de los jugadores.
     + Procedimiento ‘limpiar\_tablero’: inicializa el tablero, es decir, asignarle el caracter ‘ ‘ a cada posición del mismo.
     + Procedimiento ‘dibujar\_tablero’: muestra por pantalla el tablero para que los jugadores tengan una visión de las fichas colocadas y por colocar.
     + Función ‘tablero\_lleno’: función booleana que verificará si hay o no, disponibilidad en el tablero para colocar fichas. Esta función toma en cuenta los siguientes pasos:
       - Recorre cada celda del tablero a través de dos ‘para’ (uno para las filas del tablero y otro para las columnas), si el valor en cada una de dichas celda es distinto del carácter ‘ ’, incrementa en 1 el contador de fichas existentes en el tablero (variable entera ‘cont’).
       - Una vez recorrido todo el tablero, se verifica ‘cont’, si es igual al número de celdas del tablero, quiere decir que el tablero está lleno y por ende se le asigna ‘verdadero’ a dicha función, de lo contrario se le asigna ‘falso’.
     + Funcion ‘encontrar\_ganador’: función booleana que recorre horizontal, vertical y diagonalmente el tablero, para verificar si existe una consecución de cuatro o más fichas correspondientes a las colocadas por el jugador de turno (la posición del tablero en donde dicho jugador colocó su ficha, es guardada por las variables enteras ‘x’ e ‘y’, que almacenan la fila y la columna respectivamente). Esta función toma en cuenta los siguientes pasos:
       - Recorre toda la fila por medio de un ‘para’, si en cada posición encuentra el caracter ‘A’, le suma 1 a la variable ‘Acons’ y a la variable ‘Bcons’ le asigna 0; si encuentra el caracter ‘B’ le suma 1 a la variable ‘Bcons’ y a la variable ‘Acons’ le asigna 0; si una de las dos variables( ‘Acons’ y Bcons) en mayor o igual a cuatro, se le asigna a la variable ‘encontrado’(declarada en esta función), se le asigna verdadero; si dicha posición es vacío, tanto a ‘Acons’ como a ‘Bcons’ les asigna 0.
       - Adicionalmente, se recorre de manera vertical y diagonal (de arriba hacia abajo y viceversa). Cabe destacar que para estos tres recorridos restantes se aplica el mismo método para encontrar las consecuciones aplicado en el recorrido horizontal.
       - Una vez terminado todos los recorridos, a la función se le asigna lo que contenga la variable ‘encontrado’ (tomando en cuenta que dicha variable antes de los cuatro recorridos se le asignó por efecto, el valor ‘falso’).
     + Procedimiento ‘perfil\_jug1’: se encarga de actualizar el nombre del jugador 1. Solamente toma en cuenta que cuyo nombre no sea igual al del jugador 2.
     + Procedimiento ‘perfil\_jug2’: se encarga de actualizar el nombre del jugador 2. Solamente toma en cuenta que cuyo nombre no sea igual al del jugador 1.
     + Procedimiento ‘jugar’: se encarga de emular el juego “Cuatro en línea”. Toma en cuenta los siguientes pasos:
       - Se implementa una estructura iterativa (‘mientras’), la cual es controlada por la variable booleana ‘salir\_partida’(cabe destacar que para ingresar a dicha iteración, a la variable previamente mencionada, se le asignó ‘falso’), con el fin de que al momento de terminar una partida, solicitar al usuario, si desea jugar otra.
       - Se muestra el tablero, invocando el procedimiento ‘dibujar\_tablero’.
       - Dependiendo del valor de la variable ‘cont\_turn’ se muestra quién es el jugador de turno, si es impar, juega el jugador 1, de lo contrario, juega el otro jugador.
       - Se le solicita al jugador de turno el número de la columna, el cual hay que considerar que debe ser un valor numérico situado en el rango de las columnas del tablero y que exista espacio disponible para colocar fichas en dicha columna. Todo esto se maneja por medio de una estructura iterativa ‘mientras’, respaldada por la variable booleana ‘validar\_turno’, dicha estructura otras dos iteraciones: una para validar que sea numérico y situado en el rango de columna, y otra que verifique si existe espacio disponible dentro de la columna ingresada por el jugador de turno.
       - Una vez validado el turno del jugador en curso, se invoca la función encontrar ‘encontrar\_ganador’, si la misma arroja verdadero, se muestra el tablero, se le suma 1 a la variable que maneja las partidas jugadas (‘tot\_jug’) se muestra el nombre del ganador (dependiendo del valor de la variable entera ‘cont\_turn’ explicada previamente, si es impar, se incrementa en uno la variable ‘gan’, de lo contrario, se le suma 1 a la variable ‘gan2’), se invoca el procedimiento ‘limpiar\_tablero’ y se pregunta al usuario si desea iniciar otra partida (validando que teclee ‘s’ si desea jugar otra o ‘n’ si desea salir de la estructura iterativa controlada por la variable ‘salir\_partida’), si tecleó ‘n’, se le asignará a la variable ‘salir\_partida’ falso para terminar la iteración; en caso de que no encuentre ganador y el tablero este lleno (invocación de la función ‘tablero\_lleno’), se considerará un empate, se incrementa en uno las partidas jugadas (variable ‘tot\_jug’), se mostrará el tablero lleno a través de la llamada a la procedimiento ‘dibujar\_tablero’ y se pregunta al usuario si desea iniciar otra partida (validando que teclee ‘s’ si desea jugar otra o ‘n’ si desea salir de la estructura iterativa controlada por la variable ‘salir\_partida’), si tecleó ‘n’, se le asignará a la variable ‘salir\_partida’ falso para terminar la iteración. Si el jugador de turno, no ha ganado con la ficha que recién colocó y queda espacio disponible en el tablero para colocar fichas, se le suma 1 a la variable ‘cont\_turn’ para darle el turno al otro jugador.
     + Algoritmo principal: es en donde se le da la bienvenida al usuario a la aplicación. Se toma en cuenta los siguientes pasos:
       - Primero se inicializan las variables de las partidas jugadas, las ganadas por el jugador 1 y por el jugador 2 (‘tot\_jug’, ‘gan’ y ‘gan2’ respectivamente).
       - Se ingresa a una estructura iterativa ‘mientras’, controlada por la variable booleana ‘salir\_apl’.
       - Posteriormente, se presenta el abanico de opciones al usuario: 1) si desea actualizar el nombre del jugador 1; 2) si desea actualizar el nombre del jugador 2; 3) si desea jugar y 4) si desea ver las estadísticas (se requiere de una validación ara la opción que ingrese el usuario para que no sea distinta de las previamente mencionadas, para ello, se controla a través de una estructura iterativa ‘mientras’).
       - Una vez validada la opción se analiza la misma a través de de una estructura condicional ‘caso’: si la opción es 1, se llama el procedimiento ‘perfil\_jug1’; si la opción es 2, se llama el procedimiento ‘perfil\_jug2’; si la opción es 3, se llama el procedimiento ‘jugar’ y si la opción es 4, se llama el procedimiento ‘estadisticas.
       - Una vez que el usuario salga de alguna de las subrutinas previamente dichas, se le pregunta al mismo si desea salir de la aplicación (validando que teclee ‘s’ si desea salir de la estructura iterativa ‘mientras’ controlada por la variable ‘salir\_apl o ‘n’ si desea elegir otra opción del abanico) si tecleó ‘s’, se le asignará a la variable ‘salir\_apl’ verdadero para terminar la iteración.