**Juego JavaScript:**

Este juego consiste en un tablero 10x10, en el que se encuentra un héroe (Rayo Maqueen) y debe encontrar un tesoro (Mate), de esto se encargará el jugador. El jugador deberá tirar un dado y el número que le salga con la tirada, será el numero de posibles movimientos vertical o horizontalmente en el tablero.

**Requisitos previos:**

* Tener un entorno que soporte JavaScript (como un navegador web con la consola abierta o un editor de código compatible con Node.js).
* Conocimientos básicos de JavaScript (variables, funciones, control de flujo).

**Funciones principales:**

1. **iniciarJuego():**

* Es la función principal que controla el flujo del programa.
* Llama a otras funciones como validarNombre(), generarTablero(), colocarHeroe(), etc,…
* Esta función se carga después de cargar todo el documento HTML, de esta manera evitamos errores.

1. **validarNombre():**
   * Esta función se encarga de verificar si el nombre introducido por el usuario es válido o no. Este nombre debe condicir ciertas condiciones:
     1. Un nombre que contenga al menos 4 letras.
     2. El nombre no debe contener numeros.
   * Si el usuario no cumple estas condiciones se le mostrará un mensaje con el error concreto que ha cometido.
2. **GenerarTablero():**
   * Esta función se encarga principalmente de generar el tablero de juego.
   * También esconde el formulario para no dejarlo visual al usuario y muestra el contenedor que contiene el tablero de juego.
   * La tabla se genera dinámicamente mediante un ciclo for, además se da un id único a cada tabla, de esta manera permite conocer la posición del héroe en todo momento.
3. **colocarHeroe():**
   * Función encargada de colocar el héroe en las celdas correspondientes.
   * Genera una etiqueta “img”, y le añado la ruta de la imagen, esta imagen representará al héroe.
4. **tirarDado():**
   * Función encargada de permitir al usuario tirar el dado.
   * Esta función se añade en un evento de escucha el cual captura cuando el usuario haga click en el botón tirar dado, cambiara la imagen del dado con el numero representante de la tirada.
5. **resaltarMovimientos():**
   * Esta función resalta las celdas a las que el usuario tiene permitido realizar movimientos con color rojo.
   * Se resaltarán las celdas acordes al número obtenido con la tirada del dado.
6. **moverHeroe():**
   * Función encargada principalmente de mover el héroe a la celda seleccionada por el usuario.
7. **comprobarRanking():**
   * Esta función compara el numero de tiradas realizadas por el usuario y las compara con el recordActual del ranking, utilizando localStorage para almacenar el record de menor número de tiradas.
   * Una vez el usuario termine de ver el ranking, se le mostrara una despedida.
8. **terminarJuego():**
   * Esta función esconde el tablero y muestra un mensaje de despedida al usuario en pantalla.

**Posibles “trampas”:**

* Si el usuario realiza un movimiento e intenta realizar otro movimiento sin tirar el dado, le saltara un alert donde se le especifica que debe tirar el dado para realizar un nuevo movimiento.
* El usuario si saca un numero bajo y quiere tirar el dado sin realizar movimiento, no se le permitirá, ya que el botón de tirar dado se oculta hasta que realice el movimiento.

**Desafíos encontrados:**

* **Mostrar las imágenes dentro de las celdas.** La imagen del tesoro la intentaba introducir como una imagen dentro de la celda, y al comprobar los elementos del HTML en consola no cargaba, aunque no daba ningún error en el código. Al final opte por ponerla como imagen de fondo en la celda.
* **Controlar que el jugador no haga más de un movimiento por tirada.** El jugador podía realizar el movimiento y si quería recapacitar su tirada, tenia la posibilidad de mover el héroe a una nueva posición. Para solucionar este error, hice una variable de control boolean y condicione a que si esa variable está en false, el jugador no puede mover el héroe.
* **Mostraba varias imágenes del dado por tirada.** En cada tirada las imágenes del dado se acumulaban, saliendo los resultados del dado de las anteriores tirada y de la actual tirada. Para solucionar este error hice una condición, en la que si existe un dado en el contenedor, se cambiase por la nueva imagen del dado y si no existía imagen del dado simplemente se añadia.