Pseudocódigo del Juego Memotest

1. Inicialización y Preparación del Tablero

1. **DEFINIR** `arrayColores` con 24 colores, donde cada color está duplicado (12 pares, 24 piezas en total). *Ejemplo: ["green", "blue", ..., "white", "white"].*
2. **MEZCLAR** `arrayColores` usando una función de mezcla (`_.shuffle` en el código). **ASIGNAR** el resultado a `piezasMezcladas`.
3. **OBTENER** el elemento contenedor del tablero (`#container-memotest`).
4. **PARA CADA** color en `piezasMezcladas`:
    - **CREAR** un nuevo elemento `div` (la "pieza").
    - **ASIGNAR** el color como un atributo de datos (`data-color`).
    - **AÑADIR** las clases CSS `pieza` y `background-pieza` (para mostrar la imagen de cubierta).
    - **AÑADIR** la pieza al contenedor del tablero.

---

2. Variables de Control del Juego

1. **OBTENER** todas las piezas creadas y **ASIGNAR** a `piezas`.
2. **INICIALIZAR** `bloqueo` a **Falso** (controla si se pueden hacer clics).
3. **INICIALIZAR** `primeraSeleccion` a **Nulo** (almacena la primera pieza volteada).
4. **INICIALIZAR** `segundaSeleccion` a **Nulo** (almacena la segunda pieza volteada).
5. **INICIALIZAR** `turno` a **1**.
6. **INICIALIZAR** `contadorJugador1` a **0**.
7. **INICIALIZAR** `contadorJugador2` a **0**.
8. **OBTENER** los elementos HTML para mostrar:
    - `spanJugador1`
    - `spanJugador2`
    - `spanTurnoDe`

---

3. Lógica Principal: Manejo del Clic en las Piezas

1. **PARA CADA** `pieza` en `piezas`:
    - **ASIGNAR** una función de evento `onclick` a la `pieza`.
    - **SI** `bloqueo` es **Verdadero**, **RETORNAR** (ignorar el clic).
    - **VOLTEAR** la `pieza`:
        - **ESTABLECER** el color de fondo al valor de su `data-color` (muestra el color).
        - **ELIMINAR** la imagen de fondo (oculta la cubierta).

- **SI** `primeraSeleccion` es **Nulo**:
  - **ASIGNAR** la pieza actual a `primeraSeleccion`.
  - **RETORNAR**.
- **SINO** (`primeraSeleccion` ya tiene valor):
  - **ASIGNAR** la pieza actual a `segundaSeleccion`.
  - **ESTABLECER** `bloqueo` a **Verdadero** (previene más clics).
  - **LLAMAR** a la función **EvaluarSelección**.

---

4. Evaluar selección

1. **SI** `primeraSeleccion.data-color` es **IGUAL** a `segundaSeleccion.data-color` (Coincidencia):
   - **LLAMAR** a **Contador(turno)** para sumar 1 al marcador del jugador actual.
   - **DESACTIVAR** los eventos `onclick` de `primeraSeleccion` y `segundaSeleccion` (las piezas quedan volteadas).
   - **LLAMAR** a **ResetTurno**.
   - **ESTABLECER** `bloqueo` a **Falso**.
2. **SINO** (No Coincidencia):
   - **ESPERAR** 1.5 segundos (`setTimeout`):
     - **VOLVER a CUBRIR** `primeraSeleccion` y `segundaSeleccion`:
       - **QUITAR** el color de fondo.
       - **AÑADIR** la clase `background-pieza` (muestra la imagen de cubierta).
     - **LLAMAR** a **ResetTurno**.
     - **LLAMAR** a **CambiarTurno**.
     - **ESTABLECER** `bloqueo` a **Falso**.

---

5. Funciones Auxiliares

FUNCIÓN: ResetTurno

1. **ESTABLECER** `primeraSeleccion` a **Nulo**.
2. **ESTABLECER** `segundaSeleccion` a **Nulo**.

FUNCIÓN: CambiarTurno

1. **SI** `turno` es **1**:
   - **ESTABLECER** `turno` a **2**.
2. **SINO**:
   - **ESTABLECER** `turno` a **1**.
3. **ACTUALIZAR** el contenido de `spanTurnoDe` con el nuevo `turno`.

FUNCIÓN: Contador(turnoActual)

1. **SI** `turnoActual` es **1**:
    - **INCREMENTAR** `contadorJugador1` en 1.
2. **SINO**:
    - **INCREMENTAR** `contadorJugador2` en 1.
3. **ACTUALIZAR** los marcadores en el HTML:
    - `spanJugador1.textContent` = `contadorJugador1`.
    - `spanJugador2.textContent` = `contadorJugador2`.
4. **VERIFICAR** condición de **FIN DE JUEGO**:
    - **SI** `contadorJugador1` es **6**:
        - **MOSTRAR** alerta "GANO EL JUGADOR 1".
        - **RECARGAR** la página.
    - **SINO SI** `contadorJugador2` es **6**:
        - **MOSTRAR** alerta "GANO EL JUGADOR 2".
        - **RECARGAR** la página.