1) Define required constants:
    1.1) Define a colors object with keys of 'null' (when the block is not in play), and blocks 1, 2, 3, 4. The value assigned to each key represents the color to display for an empty block (null), and the blocks when they are flashing.
    1.2) Define a sound object with keys of null (when the sound is not firing, and blocks 1, 2, 3, 4

2) Define required variables used to track the state of the game:
    2.1) Use a board array to represent the blocks.
    2.2) Use a loser variable to represent the two different possibilities - player lost, or game in play.
    2.3) Ignore clicks while the pattern is playing
    2.4) Timing variables so that it can be adjusted every few levels

3) Store elements on the page that will be accessed in code more than once in variables to make code more concise, readable and performant:
    3.1) Store the 4 elements that represent the blocks on the page.
    3.2) tag line
    3.3) replay button

4) Upon loading the app should:
    4.1) Initialize the state variables:
        4.1.1) Initialize the board array to 4 nulls to represent the blocks where the player will need to click. The 4 elements will "map" to each block, where index 0 maps to the left most block and index 3 maps to the right most block.
        4.1.2) Initialize loser to null to represent that the player has not lost yet.
    4.2) Render those state variables to the page:
        4.2.1) Render the board:
            4.2.1.1) Loop over each of the 4 elements that represent the blocks on the page, and for each iteration:
            4.2.1.1.2) Use the index of the iteration to access the mapped value from the board array.

4.3.1.1.3) Set the background color of the current element by using the value as a key on the colors lookup object (constant).

4.2.2) Render a message:

4.2.2.1) If loser has a value other than null (game still in progress), use asynchronous function to flash the colors and sounds of the blocks in a time interval

4.2.2.2) Otherwise, render a losing message to the player

4.3) Wait for the user to click a block, before the first color fires off

4.4) render the asynchronous function that controls the block’s color and sound

5) Handle a player clicking a block:

5.1) Obtain the index of the block that was clicked by either:

5.1.1) "Extracting" the index from an id assigned to the element in the HTML, or

5.1.2) Looping through the cached block elements using a for loop and breaking out when the current square element equals the event object's target.

5.2) If the block at the index has not fired, immediately return, and the player loses.

5.3) If loser is not null, immediately return because the game is over.

5.4) Show the replay button, once the game is lost

5.5) All state has been updated, so render the state to the page .

6) Handle a player clicking the replay button:

6.1) Do steps 4.1 (initialize the state variables) and 4.2 (render).