Title: Listen Up!

Logline: Play it by ear!

Joseph Swetz
Login: jsswetz

Play Description

A sequence of sounds will be given to the player. Each sound corresponds to a numbered bead on the grid, amongst many other sounds. Clicking on a bead will play its corresponding sound. The goal is to figure out which beads, when played in a particular order, will recreate that sequence. At anytime, the player can press the enter key to hear the sequence again. The game will consist of three levels, each one more difficult than the previous.

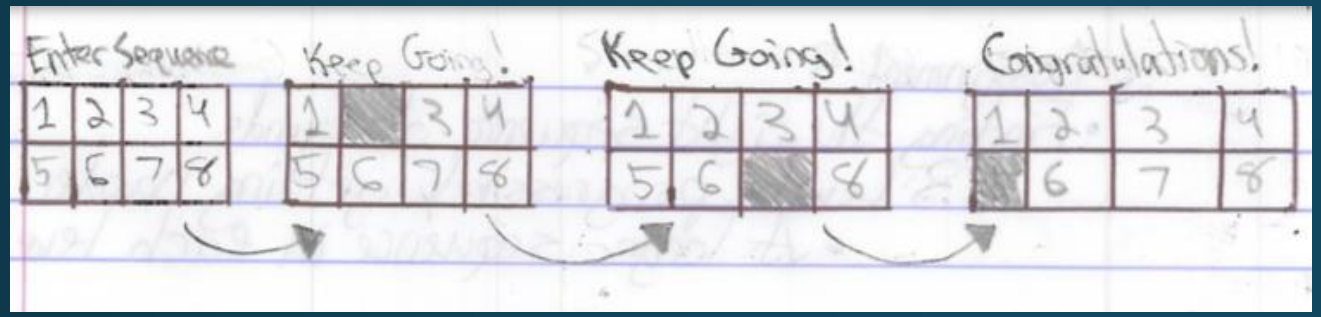
Level 1: Solve a sequence of 3 sounds from a grid of 8 beads.
The sounds will be very discernable from one another.

Level 2: Solve a sequence of 4 sounds from a grid of 12 beads.
The sounds will come from two instruments, so it will be easy to tell some sounds apart, but harder for others.

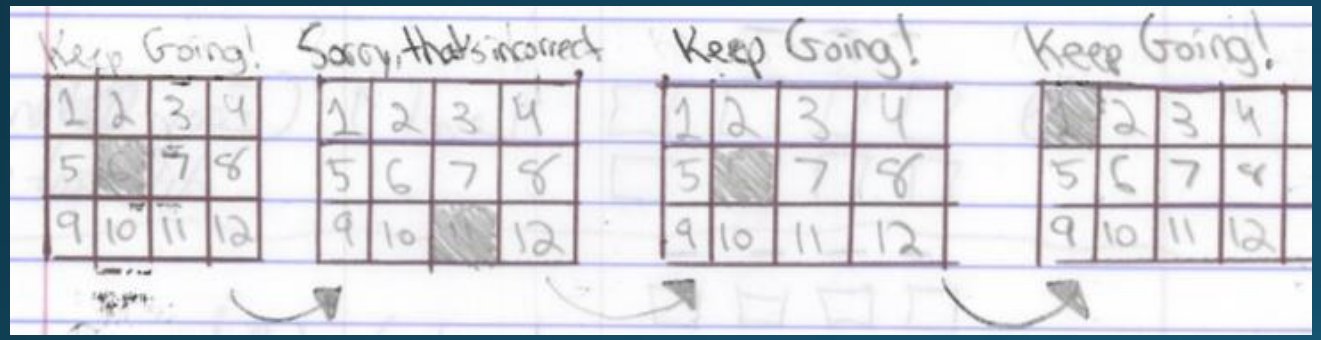
Level 3: Solve a sequence of 5 sounds from a grid of 16 beads.
The sounds will all be notes on a piano, making it tricky to figure out which note is which.

Screen Mockups

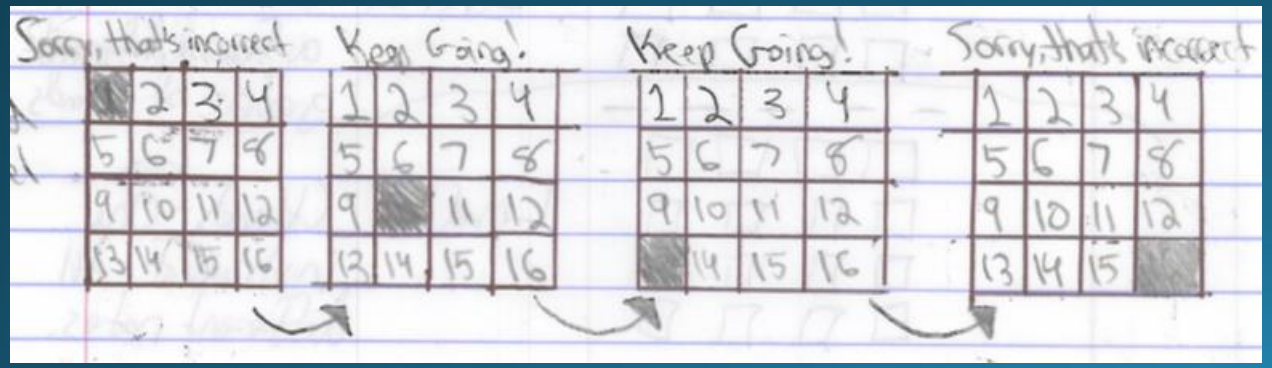
Level 1:



Level 2:



Level 3:



Key Features

- The sequence that players must solve and the sounds given to each bead must be random with every new game.
- Multiple levels, each with increasing difficulty.
  - Allows players to get a handle on how the game will work.
- The player must be able to replay the sequence that they're trying to solve at any time for reference.
- A way to tell the player that they're making progress.
  - A message on the status line telling the player that they're going in the right direction.
    - Or instead turn the grid shadow to green when they play a series of sounds from the sequence so they know they're on the right track.

Demo Code: Board Initialization

```
//Set the board up for 8 different sounds, and determine the winning sequence out of them.
initializeLevel1() {
    numberBeads();//Number the beads from 1 - (BOARD_WIDTH * boardHeight)
    loadAllAudio();//Load up all necessary audio files.
    boardSounds = [];//Use to keep track of the sounds already on the board.
    for (var y = 0; y < boardHeight; y += 1) {
        for (var x = 0; x < BOARD_WIDTH; x += 1) {
            var nextSound = getRandomSound(boardSounds, level1Library);//Generate random sound.
            PS.data(x, y, nextSound);//Place in corresponding bead.
            boardSounds.push(nextSound);//Add to running list of all sounds in the board.
        }
    }
    winningSequence = makeWinningSequence();//Determine the winning sequence.
},
```