Process & Decision Documentation

Project/Assignment Decisions

I originally wanted to have it so that when the blob hit the right edge of the canvas, it would reset and change my “level” text banner accordingly to go up by one each time, but due to technical limitations I couldn’t figure out a way to make this work. However, I still kept the level border idea and used a ‘for’ loop to execute that piece of text to show up and increase by 100px each time.

Role-Based Process Evidence

Although there is no process evidence I can show (besides my GenAI documentation), I can state that the level border idea was completely thought of from scratch. I had reviewed p5.js videos to get a review of how loops worked and asked GenAI to make a working loop which I understood immediately.

I did use GenAI for creating a working loop.

GenAI Documentation

Date Used: February 10, 2026

Tool Disclosure: ChatGPT 5.2

Purpose of Use: I used GenAI for helping me create a working for loop with an idea I had in mind of a level border.

Summary of Interaction: The tool helped me in creating a working for loop. I only needed one singular interaction since I know how loops work in p5js. I just needed extra clarification on whether or not the text would be looped properly.

Human Decision Point(s): GenAI was able to help me for what I needed without any extra outputs, I didn’t overrule anything.

Integrity & Verification Note: The only way I could verify GenAI’s outputs was what I needed for myself. I had reviewed several p5.js videos on loops before completing this sidequest and with the outputs GenAI provided me, I understood exactly what the loop was doing and how it worked. From personal verifications, I deemed it as good and fair and I understood everything it provided me, I just needed help with starting the idea but I understand it fully.

Scope of GenAI Use: GenAI provided me with a working loop, however, I changed certain variables to be more specific to what I've learned in the past and what I was aiming to achieve. In CS105 and CS106 we were taught it was good practice to set variables within a loop as 'i' or 'j' so as to not confuse us from 'x' and 'y' which is a change I made. I changed the text's 'y' positioning to be 75 and the 'i' variable to 40 to start the loop closer to the left on the x-axis. I also ensured the 'word' variable was changed to fit my need, changing it to 'Level 1 -". I also decided to change the 'word' variable from a single word to an array of words that can be picked out.

Limitations or Misfires: The tool worked exactly as I needed it to. No limitations or misfires.

Appendix

(My input prompts (not the code I pasted in) are highlighted for better navigation.)

using a loop ONLY IN SKETCH.JS is it possible to meet any of these requirements: - use loops to dynamically place tiles, obstacles, or words don't do anything yet just tell me if it's possible (then pasted the code below)

/* Week 4 — Example 5: Example 5: Blob Platformer (JSON + Classes) Course: GBDA302 Instructors: Dr. Karen Cochrane and David Han Date: Feb. 5, 2026 This file orchestrates everything: - load JSON in preload() - create WorldLevel from JSON - create BlobPlayer - update + draw each frame - handle input events (jump, optional next level) This matches the structure of the original blob sketch from Week 2 but moves details into classes. */ let data; // raw JSON data let levelIndex = 0; let world; // WorldLevel instance (current level) let player; // BlobPlayer instance function preload() { // Load the level data from disk before setup runs. data = loadJSON("levels.json"); } function setup() { // Create the player once (it will be respawned per level). player = new BlobPlayer(); // Load the first level. loadLevel(0); // Simple shared style setup. noStroke(); textFont("sans-serif"); textSize(14); } function draw() { // 1) Draw the world (background + platforms) world.drawWorld(); // 2) Update and draw the player on top of the world player.update(world.platforms); player.draw(world.theme.blob); // 3) HUD fill(0); text(world.name, 10, 18); text("Move: A/D or ←/→ • Jump: Space/W/↑ • Next: N", 10, 36); } function keyPressed() { // Jump keys if (key === " " || key === "W" || key === "w" || keyCode === UP_ARROW) { player.jump(); } // Optional: cycle levels with N (as with the earlier examples) if (key === "n" || key === "N") { const next = (levelIndex + 1) % data.levels.length; loadLevel(next); } } /* Load a level by index: - create a WorldLevel instance from JSON - resize canvas based on inferred geometry - spawn player using level start + physics */ function loadLevel(i) { levelIndex = i; // Create the world object from the JSON level object. world = new WorldLevel(data.levels[levelIndex]); // Fit canvas to world geometry (or defaults if needed). const W = world.inferWidth(640); const H = world.inferHeight(360); resizeCanvas(W, H); // Apply level settings + respawn. player.spawnFromLevel(world); }

Yes — it is definitely possible. ✅

Even without modifying your classes or JSON data, you can use a loop in **sketch.js** (in `setup()` or `draw()`) to dynamically place things like:

- Tiles or platforms in a grid pattern
- Obstacles (spikes, blocks, etc.) along the floor or walls
- Words or text repeated across the screen

The key is that loops can generate positions or iterate over arrays, and you can call drawing functions (like `rect()`, `ellipse()`, or `text()`) inside the loop.

If you want, I can show a **small example of how to do this with your current setup**, just using loops in sketch.js. Do you want me to?

okay lets take it slow, try doing the words thing first

Perfect! Let's start simple and build it step by step.

Here's one way you could **dynamically place words using a loop in `sketch.js`**, without touching your `WorldLevel` or `BlobPlayer` classes. You could add this inside `draw()` (or `setup()` if it's static text):

```
function draw() {
  // 1) Draw the world
  world.drawWorld();
  // 2) Update and draw the player
  player.update(world.platforms);
  player.draw(world.theme.blob);

  // 3) HUD
  fill(0);
  text(world.name, 10, 18);
  text("Move: A/D or ←/→ • Jump: Space/W/↑ • Next: N", 10, 36);

  // 4) Dynamic words example
  fill(255, 0, 0); // red words
  let word = "BOOM";

  // Loop to place the word multiple times horizontally
  for (let x = 50; x < width; x += 100) {
    text(word, x, 100); // y = 100 for now
  }
}
```

✅ What this does:

- Loops from `x = 50` to the canvas width, stepping by 100 pixels each iteration
- Draws the word `"BOOM"` at each `x` position
- You can change `y` to stack rows or make it dynamic

If you want, the **next step** can make it even cooler by:

- Looping **both horizontally and vertically**
- Picking words from an array so it's not the same word every time

Do you want me to do that next?

if i were to make a word as an array and then put into the text: text(word[1], x, 100); that would still work right and then just take the second value on the array?

Yes — exactly! ✅

If you define `word` as an array, like this:

```
let word = ["Level 1 -", "Level 2 -", "Level 3 -"];
```

Then you can access a specific element by its index. For example:

```
text(word[1], x, 100);
```

- `word[0]` → `"Level 1 -"`
- `word[1]` → `"Level 2 -"` ✅
- `word[2]` → `"Level 3 -"`

So yes, it will draw **only the second element** of the array at that position.

You can also use a loop and change the index each iteration to draw different words.

If you want, I can show a **tiny example that cycles through a 3-word array across the screen**, keeping it super simple. Do you want me to do that?