

Process & Decision Documentation: Side Quest Week 4

Project/Assignment Decisions

For this Side Quest, I focused on generating a level using arrays and loops rather than manually drawing elements. I chose a small tile-based layout with simple movement and collision so that I could concentrate on understanding how level data controls what appears on screen. I also added automatic level progression to reinforce how state and reset logic work across multiple levels.

Role-Based Process Evidence

The project was built around a data-driven approach, where each level is defined as a 2D array and rendered using nested loops. Player movement and collision are handled by checking values in the grid, and level transitions reset the grid and player position to avoid leftover state issues. The sketch was play tested to confirm that movement, collisions, and level switching worked correctly without breaking.

GenAI Documentation

Date Used: February 9th, 2026

Tool Disclosure: ChatGPT and GitHub Copilot (Visual Studio Code)

Purpose of Use:

Brainstorming a simple Week 4-appropriate idea, clarifying how arrays and loops could be used to generate a level, and checking basic p5.js syntax and logic while coding.

Summary of Interaction:

GenAI tools were used to support brainstorming, clarification of course concepts, and incremental refinement of the code structure. They helped me think through how to represent a level as array data, how loops could be used to draw tiles dynamically, and how to reset player state correctly when switching levels, while keeping the project intentionally low effort.

Human Decision Point(s):

- Deciding to use arrays instead of JSON to reduce setup complexity and avoid file-loading issues.
- Choosing a very small grid and limited tile types to keep the Side Quest manageable.

- Implementing a grid copy when loading levels to prevent accidental modification of the original data.
- Simplifying visuals and interactions to avoid overbuilding beyond the scope of a weekly exercise.

Integrity & Verification Note:

All GenAI suggestions were tested directly in the browser to ensure they worked as expected. I verified that level transitions reset player position correctly and that no state carried over between levels. When suggestions felt too complex or unnecessary, they were simplified or discarded to better match course expectations.

Scope of GenAI Use:

GenAI did not determine the final design, level layouts, or interaction rules. The level data, visual style, and final logic decisions were created and adjusted by me, with GenAI acting as a support tool rather than an author of the final submission.

Limitations or Misfires:

Some GenAI suggestions initially introduced more structure than needed for a Side Quest. These were scaled back to maintain clarity and avoid common errors such as broken reset logic or unnecessary abstractions.

Appendix

User:

I'm reading the Week 4 instructions and I see we need to generate a level using arrays or JSON. I'm a bit unsure how complex this is supposed to be. Is a really small grid okay?

GenAI:

Yes, a small grid is completely fine. You can represent the level as a 2D array where each number maps to a tile type, then use nested loops to draw everything. That already meets the requirement.

User:

Okay, that helps. I don't really want to deal with loading files if I don't have to. Can I just keep the arrays directly in the sketch?

GenAI:

Yes. Using arrays directly inside your sketch is acceptable and often easier for a weekly Side Quest. JSON is optional unless the prompt specifically requires it.

User:

I started doing that, but my TA mentioned a lot of people had issues with reset logic. I'm worried about breaking things when switching levels

GenAI:

That's a common issue. A good approach is to avoid modifying your original level data. Instead, copy the grid each time you load a level and reset the player position based on the start tile.

User:

Got it. I also want walls to actually block the player. Right now it kind of just moves everywhere

GenAI:

Before moving the player, check the value of the next tile in the grid. If it's a wall tile, you simply don't update the player's position.

User:

Once the player reaches the goal, do I need a whole new screen or state for that?

GenAI:

Not necessarily. You can automatically load the next level when the player reaches the goal. That still demonstrates level progression and also satisfies the bonus requirement.

User:

Looking at this again, I feel like I might be overdoing it. These Side Quests are supposed to be low effort

GenAI:

In that case, keep the visuals simple and focus on the core learning goal: arrays and loops. You can remove extra features and just make sure movement, collision, and level switching work reliably.

User:

Okay, yeah. I'll keep it minimal and just make sure nothing breaks

GenAI:

That's a good plan. A quick playtest to confirm movement, collisions, and resets will help catch most issues before submission.

User:

I think something's wrong with my level reset. When I finish the level and it loads the next one, the player sometimes starts in a weird spot or the level doesn't feel fresh

GenAI:

That usually happens when the original level array is being modified. Are you changing the grid directly when you place the player or move through the level?

User:

Yeah, I think I am. I'm changing the start tile to a floor tile after placing the player

GenAI:

That explains it. You should keep the original level data unchanged and create a copy of the grid each time you load a level. That way, every level starts from a clean state.

User:

Okay, I tried copying the grid when the level loads, and that seems to fix it. The player now resets properly and the level feels consistent

GenAI:

Great. That's the correct approach and it also addresses the reset logic issue your TA mentioned. Just make sure you always reload from the original data when switching levels.