Name: Karen Wu, 21020152
Assignment: Individual SideQuest W5

Goal of Work Session:
The objective was to refine the technical implementation of a world-space exploration environment in p5.js. Specifically, I aimed to fix coordinate-system errors regarding UI positioning, implement a randomized procedural generation system for obstacles, and resolve a visual rendering bug (a stray black transparent box) interfering with the user interface.

Tools, Resources, or Inputs Used
- **GenAI Tools:** Gemini AI
- **Lecture Notes:** GBDA 302 course materials, W5 Example Code
- **External references:** p5.js Web Editor and official CDN documentation.

GenAI Disclosure
Date Used: February 23, 2026
Tool Disclosure: Gemini AI
Purpose of Use: Troubleshooting script linking

Human Decision Point(s):
- UI/UX Layering: I decided to move the instructional text and coordinate display outside of the push() and pop() matrix. While the AI suggested ways to calculate the text's position within the world, I chose to render it in "Screen Space" instead. This ensures the UI stays fixed to the user's viewport, preventing it from sliding off-screen as the player moves.
- Collision Detection Strategy: I opted for a "Point-in-Rect" collision check rather than a complex "Circle-to-Rect" physics library. Given the top-down nature of the Sidequest, this simplified logic was a deliberate choice to maintain high performance and readable code while still providing a solid "wall" effect.
- Procedural Generation vs. Design: I decided to transition from the AI's initial patterned obstacle placement (using modulo math) to a fully randomized initObstacles() function. I chose this to increase replayability, ensuring that every time the canvas is refreshed, the "world" feels different for the player.
- Visual Polish & Clarity: I decided to remove the "black transparent box" appearing on the canvas. I felt it cluttered the visual hierarchy and distracted from the primary goal of navigating the blue player through the grey obstacles.

Scope of GenAI Use: GenAI was used for UI coordinate troubleshooting, procedural logic generation, and visual debugging. It specifically helped bridge the gap between "World Space" (where the player and obstacles exist) and "Screen Space" (where the UI text remains fixed), while also refactoring the obstacle initialization from a static pattern to a randomized distribution.

Summary of Process (Human + Tool)
Human: Provided the initial boilerplate for player movement, camera lerp(), and the isColliding function.

Tool: Identified that the UI text was placed inside the push()/pop() matrix, causing it to "drift" with the camera.
Human: Decided to move the UI rendering after the pop() function to lock it to the screen.
Tool: Suggested a refactor for initObstacles() using random() to replace the previous pattern.
Human/Tool Collaboration: Investigated the source of a "black transparent box" artifact on the canvas, eventually identifying and removing an unnecessary rect() call in the UI layer.

Appendix: GenAI Interaction Summary

Full Conversation Link: Conversation with GenAI SideQuest 5
Transcript Summary:
The interaction focused on resolving coordinate system conflicts and enhancing the environment generation for a top-down exploration prototype. We identified a "transformation mismatch" where static UI elements were being translated by the camera's x and y offsets. We also transitioned the project from a predictable grid of obstacles to a randomized procedural layout to improve gameplay variety.

Key points of the interaction included:

- Coordinate Space Separation: Differentiating between the translated world (where the player wanders) and the static overlay (where instructions and coordinates live).
- Procedural Generation: Implementing random(0, WORLD_W) within the initObstacles loop to create a more organic, less grid-like environment.
- Graphic Debugging: Locating a stray "black transparent box" which was caused by a leftover fill() and rect() command intended for a UI background that was no longer needed.
- Input Normalization: Discussing the movement logic (using dx/len) to ensure the player doesn't move faster when travelling diagonally.