Aliyah Renomeron-Ward

CART 253

Section AA

Post Reflection Essay

Fall 2025

Instructor: Sabine Rosenberg

When this course first started, my relationship with programming was basically, “I admire it and I’m learning, but it scares me a little.” I had the vibes, aesthetics, big ideas and a love for interactive art, but the actual coding part felt like a door I couldn’t really get behind.

When I started CART 253, my relationship with programming was mostly built on curiosity and a vague sense that “people who code” existed in a place slightly above normal human functioning. I had seen creative code on social media and admired it, and had even been inspired by it in my own projects. But my own technical understanding was not as deep. I had used Javascript before but only with the help of tutorials and friends who already knew it. At the beginning of the semester, I knew the vibes I wanted to create long before I even knew the syntax.

Over the course of the semester, that changed radically. The most significant shift was not just learning how to program in p5.js and JavaScript, but learning *how to think* like a programmer. Early exercises in variables and conditionals felt almost like learning grammar: not glamorous, but transformative once I understood the connections. If-statements turned out not to be mystical gates,

but simple logic checks. For-loops stopped feeling like math puzzles and became tools I could use for repetition, animation, and dynamic visuals. I began recognizing patterns. Not just in code, but in the way computational thinking structures creativity.

This shift became even more dramatic once I started building larger projects. At the beginning of the course, creating a single-screen p5 sketch felt like an accomplishment. By the end, I had built an interactive, multi-state system with loading animations, JSON-driven narrative sequences, and unique minigames, each with its own logic, triggers, failure conditions, and visuals. What once felt impossible, slowly transformed into something I could break down, understand, and execute through code.

It wasn't that the course magically removed all difficulty. Programming is still challenging, and sometimes outright frustrating. But the difference now is that I'm no longer intimidated by that frustration. I know how to debug. I know that errors are signals, not failures. I know that if something behaves unexpectedly, I can trace back through variables, conditions, or timing issues and find the culprit. In the early weeks of the course, I would often think "this is broken and I have no idea why." Now, I think, "this is broken, but I know how to start unraveling it." That mindset shift is probably the biggest change of all.

Before this course, my art brain and my coding brain lived in different universes. Art was dreamy and narrative-driven; code felt precise and math-heavy. But once I actually started using programming in my creative work, the two sides finally clicked together.

Learning how to structure a program with states was one of those moments where everything changed. Suddenly I wasn't limited to "one screen and one interaction." I could build whole experiences.

The JSON files were another gamechanger. Being able to store all my narrative text externally and have the program pull each line in? It made the whole writing process feel more natural. Almost like the code was letting the story breathe instead of trapping it in one cluttered script. It made me think like a writer and a programmer at the same time.

And then, making the minigames… that's where everything fused. I wasn't just "applying code concepts." I was designing rhythm, tension, timing and pacing.

I didn't expect to experience code as a storytelling medium, but that's exactly what happened. The technical parts kept nudging my creative decisions, and the creative parts forced me to stretch my technical skills. They kind of became co-pilots.

I also learned that even when coding fights back, it still feeds back into the creative process. When something breaks, I have to rethink how the system works, and that usually leads to better, more intentional design. Frustration and clarity became weirdly linked.

By the end of this course, I feel far closer to owning the title "creative coder." Not because I know everything, but because I now think in computational ways. I can sketch system behaviors the same way I sketch visual ideas. I can mentally map how a player might move through a game state. I understand how data structures support narrative structures, and how simple logic can generate complex results.

My understanding of creative code has shifted from "cool visuals made by mysterious programs" to "a craft involving both precision and imagination." Coding is not just typing instructions into a machine; it's designing systems that breathe, react, surprise, or challenge an audience. It's storytelling through interaction.

Looking forward, I'm excited about and interested in how creative coding intersects with installation art, projection mapping, and responsive environments. The possibilities feel bigger than they did at the start of the semester in a motivating way.

I also want to strengthen my fluency with JavaScript outside p5.js. This course gave me a strong foundation, but I'm eager to explore frameworks that allow creative coding at a larger scale. Now that programming no longer feels like an inaccessible realm, I'm excited about diving deeper.

If there's one thing I take away from CART 253, it's that creative coding is not merely a technical skill. It's a way of thinking that blends logic with art, structure with imagination, precision with narrative. This semester didn't just teach me how to write code. It reshaped how I understand my own artistic practice. I no longer see programming as a barrier between ideas and execution. I see it as the machinery that lets ideas come alive.