Post-reflection

Weini Wang

This is my first year at Concordia, and this programming course is also the very first coding class in my life. Although programming felt a bit challenging for someone who had never touched code before, this course has been incredibly interesting for me.

I learned how to use JavaScript and p5.js, and I found that p5.js is actually very simple and intuitive—especially for making games. At the beginning of the course, the code and functions felt quite easy. But when we started learning the *for loop*, things became more difficult. Overall, I realized that JavaScript isn't that hard as long as I keep practicing.

Before this course, I only knew that programs were made of instructions, and variables allowed things to change, but I had no idea how to combine these elements to make something that moves, glows, or interacts with a user. At the start of the semester, I didn't understand the purpose of p5.js at all—I thought it might be a more advanced system. Through learning, I discovered that it is a creative coding library built on JavaScript. The setup and draw functions were completely unfamiliar at first, but after deeper study, I realized how convenient they are for creating dynamic visuals. The random() function quickly became one of my favourites when I wanted shapes to move unpredictably.

I also discovered how useful console.log() is—it helped me find and fix bugs again and again. During the early exercises, I often relied on example code; by imitating and experimenting, I gradually understood the principles behind the functions. I also want to say that the course website helped me a lot. Following Pippin and Sabine's teaching materials, I developed a new understanding of programming. The concepts of if statements, JSON, and loops were like "switches" that opened a new world for me. The if statement helped me understand relationships and logic, while for loops allowed me to create repeated shapes and manage arrays.

In my final project, I applied for loops and JSON to display stars, assign moods, and manage the star array. Through these structures, I experienced the moment when "algorithms" truly became creative tools. While building the game, watching dozens of stars fall at random speeds, flicker, and disappear made me realize that code could form a dynamic visual system that feels alive.

Throughout the course, the part I became most comfortable with was understanding states and if statements. When building menus and switching between mini-games, I became used to writing things like
if (state === "game1") { game1Draw(); }
At that moment, I understood that coding is a kind of storytelling—each state is a different chapter, and switching states moves the story forward. In my final project, I

also learned how to use a switch() statement with case "menu":, which made the transitions between menus and game screens much easier and cleaner.

I began to realize that digital art is not a static canvas. With logic, it becomes something interactive. But I also clearly see the challenges I still face: making my code cleaner, shortening it, organizing logic more clearly, and debugging complex interactions. Sometimes it takes a long time to find the source of a bug. In the future, I hope to build more complex UI systems, write more efficient code, and use global variables and clear logic structures to keep my projects organized. I plan to bring these challenges into my future practice.

I now clearly understand that as an artist, learning code allows me to turn imagination into interactive artwork. Images no longer stay still on a 2D plane—they can respond, react, and create new experiences for the viewer. I can now do things I never imagined before, such as building multi-scene structures (menus, levels, end screens), controlling the movement of characters and objects (like sine-wave flight paths), handling user input, creating my own mini-games, and using JSON and arrays to manage large amounts of content. These skills not only help me understand others' code but also allow me to create through code myself.

Looking ahead, I feel that I still have some distance to go before becoming a fully mature creative programmer. Right now, I can write basic interactivity, animations, and game structures, and I understand the core logic of p5.js and JavaScript. But I also know I'm not yet skilled with more complex systems, advanced visual effects, or large-scale code organization.

Before this course, I thought creative coding was simply "making things move." Now, I understand that it's a way of combining logic, visuals, and interaction into one artistic practice. Code is not just a technical tool—it can express ideas, design experiences, and tell stories. This shift in understanding showed me that creative programmers are not just "people who write programs," but people who build artistic experiences through code.

In the future, I hope to continue learning: how to build more complete interactive storytelling structures, how to manage large-scale works, how to create more complex animations and visual systems, how to use data in creative ways, and how to design richer interactions. Most importantly, I feel excited for what comes next. I can already use code to tell stories, design interactions, and build worlds—and in the future, I hope to create digital works with more depth, poetry, and interactivity.

Although I'm still learning, I am much more confident now than I was at the start of the course, and I'm eager to continue exploring this path.