Final Self-Assessment

Rosie Wang

Unlike the first half of the semester, the second half of the semester of Creative Coding was full of challenges for me because vector and string manipulation are completely new for me to learn. Even I had used vector in my midterm project, I didn’t know it very much. Thus, I put much effort into understanding the mechanism behind the code, reading the examples again and again. However, compared with actual programming, I was almost never trapped by the concept. The first step in learning is imitation. First, I just tried to modify the original codes to achieve my goal, treating the code like a formula. As the function I needed to realize became more and more complicated, I had to create my own code. If I could write my own code, I should have mastered and remembered how to use the new function or class. However, in fact, I thought I didn't do it very well because I still needed to read the examples of a vector when I need to write a class for the moving clouds and birds. Although I spent much time to understand them, I still needed a guide to applying them. That’s part of the reason I used vector in my final project but only wrote the basic function. The other part was that the cloud and birds can disturb the player’s focus if they change too much.

As for the two programming software, Processing and P5.js, I used to prefer Processing because I felt more comfortable with Java than JavaScript. However, after using P5.js in sketch 5 and final project, I realized the convenience of using JavaScript to code. Now I am familiar with the basic structure and syntax of both two languages. If I want to use another library as assistance, I may choose P5.js because it’s easy to import a library there. Now I am proficient with P5.play library. But most of the time I still cannot run my code written in P5.js correctly in the browser. My sketch 5 code ran differently on OpenProcessing and the browser. It's even worse that I cannot run my final project in the browser until now. Thus, I wrote all my code on OpenProcessing directly if I want to use P5.js to code.

When talking about OOP and procedural programming, I always decide which one to use based on how many similar objects I need to create. OOP is designed as a classified group to have many similar objects with the same attributes and but are different in values or features. It uses a class as the form for this group. If I want to create many moving shapes with same attributes such as position, velocity, and acceleration, I can build a class, write all these attributes for each object in this group, and assign different values to differentiate the different shapes. If I don't use OOP to create those shapes, I should write those attributes and assign them values multiple times. For procedural programming, the main purpose is to divide a problem into different procedures, the sequence of statements, and combine them together to form a complete code. The code we usually write in the draw function belongs to this procedural programming.

In my final project, I learned a completely new library, P5.play library, and used it to write most of the code. This should be the breakthrough I made in this project. P5.play library helped me to treat images as a sprite, an object. I read the reference to this library many times to find the function that I can use to achieve my goal. The functions like "collide" and "overlap" helped me to simplify many steps. To make my heart hit the mountain until they looked at crashing each other, I should abandon using “collide” function because the heart will hit the mountain as soon as it collided the transparent edge of the image of the mountain. After searching in the reference, I found that there’s a function called “overlapPixel” that “can be used to check a point collision against only the visible part of the sprite.” Then I replace the collide() with this function. This experience emphasizes the importance of reading the reference to find the solution. Since I have many sprites in this project, I need to make sure them to appear in order. It indeed taught me how to organize the variables.

I always had bugs in my final project, such as the flower and envelop can only appear once, and the even the mountain didn't appear, the heart can hit it. When I had bugs, I usually resolve it by myself by debugging them again and again. I create another sketch for test on OpenProcessing. After I used the comment tool to find out which part had a bug, I would cut and paste the code of this part into the test sketch and correct my code until it could run without bugs. Sometimes I had bugs because I forgot the difference between setup() and drew (). Sometimes because I put the sprites in the wrong order.

In the future, I wish I could handle different programming languages, and I'll keep learning them because I think programming is a basic skill for a college student and will improve my competitiveness in the workplace. Another reason is, I like programming and enjoy it. I will take Code of Music in IMA next semester and take Intro to Web Development in sophomore year. Coding experience will definitely help me to pass those two courses.