Lilia Torres

Project Reflection

After completing my culmination, the first thing I can say is that I feel accomplished. While I felt like my original idea and plan was straightforward, I found myself getting stuck a lot in many parts of development. I knew heading into this project that I was going to challenge myself with scripting and since programming is not one of my strengths and I have little experience with Unity, I'm proud to say that I overcame it.

When I first started the planning process, I was driven by the idea of creating something new that could help teach music to anyone who wanted to learn. I thought about its physical design and the different environments in which my game can be used, such as in schools, more specifically music classes, or in arcades for a fun experience. I wanted to create a physical prototype inspired by arcade cabinets. At the time, I felt like I had a lot of time ahead of me and with the right planning, I could complete my physical prototype.

However, once the semester began, I realized how fast time was going. I started my development by creating a Game Document to organize my ideas as well as help decide on which scripts I needed and in what order to work them. I knew I wanted my game in 3D and that I wanted to implement midi as the main input and method of control. This for the first two months helped me stay on track with my progress.

By March I was already researching midi implementation in Unity and how to install and use the MidiJack plugin. The physics in the 3D setting were working well up until the midi input was implemented. I began getting collision errors and ray casting errors where certain objects weren’t being read properly and the ball wouldn’t react the way I needed it to in game. There were numerous clashes in the script that caused object colliders to cancel out and ultimately deactivate. I found myself rewriting scripts alot throughout March and early April. Towards the end of March, I concluded that in order to stay on track I would need to scrap some of my original ideas.

One of the first changes I made was in regards to the size of my controller. I wanted a midi controller on the larger side because having more midi inputs meant the player would have a wider selection and a more challenging experience. However, it was hard finding a controller within my budget, since my plan was to get two. I ended up opting for a 49-key midi controller which at first, felt like it wasn’t enough. After receiving it and testing out the midi, it turned out the game could still be fun with 49 keys since it still has players about 4 octaves to play with.

Another change I had to make was switching my game from 3D to 2D. I realized that the scripting methods I was using to implement the midi input and the ball physics wasn’t working well in 3D. It would be better to switch to a 2D environment so I could focus on only two vectors and make sure the main colliding components were working. This was upsetting because it meant I was no longer following my game document and would need to research Unity scripting for 2D, as well as starting my assets from scratch. This resulted in me changing the aesthetic of the game early on, no longer going to a neon colorful aesthetic, but for a more cartoon-like style. I tested out some angles and still managed to keep my game at a birds-eye-view perspective.

Despite all this troubleshooting and rewriting of the scripts, I felt like I was still somewhat on track. I did fall a little bit behind schedule as April approached but I was still

making good progress. The midi implementation was giving me some trouble as some of the collisions still weren't being detected properly, but at least the game was functioning and it was possible to play one round without compiling any errors. My main approach was to create a keyboard skeleton which would double as an indicator. When a midi key gets pressed, the on screen piano skeleton would light up, indicating a successful connection. All the C chord triads on the keyboard are also connected to a collider above the keyboard, serving as the players' paddle. Whenever The player presses a C chord triad, the paddle lights up, enabling the box collider and waiting for the ball to collide in order to play the corresponding chord. If the player were to let go of the chord early or miss the ball entirely, it would result in a game over screen.

After transitioning to a 2D unity project, the only issue I came across while touching up my script was that when the player would lose and the game resets, I kept losing my sound and light up indicators on the keyboard skeleton. However, the triad chord indicators still worked and would light up accordingly. This meant there was an issue when transitioning between the game over screen and the actual level, which prevented the game scene from restarting properly. This was probably the hardest bug to try and resolve. Since i was scripting each individual midi key rather than using an array, I believed it would be more convenient to keep have one C# script manages everything midi related. It turns out the opposite happened and using the same script for multiple methods is what was causing the errors when restarting the level. Had I created independent scripts for specific functions this issue would've most likely been resolved from the start.

Overall, I feel like I learned a lot through my progress with my culmination. There were many moments of frustration and times where I felt lost with the scripting, but looking up resources online and watching videos helped me alot throughout this project. When I completed my prototype, I ended up feeling more confident in my knowledge of programming and unity scripting. I got to experience game development and the various processes that go into creating a game; asset design, sound and programming. I also learned the importance of working with an open mind and adapting to different situations. Many of the moments when my scripts weren't working required me to think of alternate approaches and methods that could give me the results I wanted, and I would need to adapt the rest of my code to match the newer methods. These are some skills I see myself using not only as I continue to develop Piano-Plunk, but in future music projects as well.

In regards to my future goals with this project, I will definitely be continuing development of Piano-Plunk. I want to create more levels using different keys and chords, as well as implement 2-player functionality.This would mean creating more chord and triad banks and creating a randomization function that would select a key and chord for the players. Additionally, updating the sound backs to play longer chords would help the game feel smoother and more melodic. I would also love to expand the sound banks entirely to allow players to select what sound they want to play as, making the game more customizable. Finally I want to implement a playback system that would allow players to playback their creation. Nonetheless, choosing this game as my culmination has taught me alot about planning and development and I'm looking forward to continuing to use these new skills on future updates of Piano-Plunk as well as in my future career.