Jonathan Phung

6/1/20

**APCSA Capstone Tutorial**

For my APCSA Capstone, I decided to make my own rendition of the puzzle game Puyo Puyo. Puyo Puyo is a puzzle game similar to Tetris in design, but different in mechanics. It is similar in how the player is given a grid and pieces that fall into it, but different in how the goal is to group same colored puyos into groups of four or greater. The group is then removed, the stack falls accordingly, and then new groups of four are popped, and so on. There are both singleplayer and multiplayer game modes, with some pitting you against your high score or other players respectively.

I started development on 5/15 and ended on 5/30, although my actual 11.5 hours of development were sporadic and concentrated due to my other schoolwork and responsibilities. For the final three days, I was also at my grandparents’ house due to my house being fumigated, and due to the different, unfamiliar setup I was using, I do not believe I was as productive as I usually was, but the circumstances were out of my control.

The final product, sadly, was not exactly what I wanted. I was hoping to be able to bring the game to life, or at least have it be functioning, but that was not what happened. Thankfully, I had developed enough to qualify for the capstone, but it still did not feel good knowing I did not meet the goal I set. In hindsight, maybe I was too ambitious with what I wanted to do. The FBLA Game Development category, or any FBLA software development category for that matter, has much more time than I did when I decided to do this project, and that kind of perspective would have served me well in determining what I should have shot for with respect to the project. Ultimately; however, I gained great insight into the application of the Java skills I developed in the class into a real project as well as learning new skills in game development with Java. It was different than anything I had ever done before, like on FRC, and I greatly appreciated the experience, if nothing else.

The timeline of my development is roughly outlined in my worklog and Github commits. Looking back, I could have been clearer in what I did and did not do, but I also felt that due to how long and hard I was working in most all of my development sessions, it was hard for me to describe what I did a majority of the time in a clear and concise way. Terms like “basic logic” and “basic graphics rendering” will be elaborated upon further here, although in future, I should be more focused in my development periods, saving and committing when I decide to work on something else, even going so far as to create extra Github branches if necessary, I did not feel the need to go so overboard for this project, but it would certainly be a good idea in future.

The development of my project started long before the end of the AP Exams. I knew from the beginning that I wanted to develop a game, and I was friends with some avid Tetris players. Tetris and Puyo Puyo has been a rivalry since the release of *Puyo Puyo Tetris* (2014), and I was deeply saddened by the great amount of Tetris clones on the internet relative to the few, if any Puyo Puyo clones. Inspired by *Nullpomino*, a Tetris clone developed in Java, I wanted to make a Puyo Puyo clone in Java, and titled my project “NullpoPuyo” accordingly.

On the first day of development, 5/15, I began researching libraries, frameworks, and engines for Java game development. After searching for Java game development tutorials, I found LibGDX and determined it to be the most accessible to me given the circumstances of the project. I was already familiar with Android Studio, found numerous quality tutorials demonstrating how to use it, determined that it had strong community support, and easily set it up myself. With that decided, I got reading, first learning about setting up the framework and the graphics required for my game primarily from [the linked article](https://www.codeandweb.com/texturepacker/tutorials/libgdx-physics). My first day primarily consisted of me following the tutorial, setting up my IDE. and creating a basic sprite sheet for my game, with a little development time going towards ensuring the game could render the sprites I provided. This day was more about learning than developing, but I had to start somewhere, and I figured that the tutorials I followed knew a little more about the process than I did. I was only three hours in.

On the second day of development, 5/24, I started developing the backend game logic. Specifically, I began working on higher level concepts, such as creating a UML for the project and determining what game processes I would need to create and how they would be represented. I also discussed the project with my good friend James, former programming captain at MORE Robotics who is now at NYU majoring in computer science with game development as a hobby. He gave me great insight into the difficulty and feasibility of the project as a whole, and gave me a few pointers on how I should implement certain systems. He mainly described how the game, at the highest level, should not be too hard to develop, as each of the game’s systems are discrete and should be able to interact with each other without issue, at least with how he described it. One of the tips he gave me was to represent the 6 x 12 game board as a 2D array and surround the board itself with nonpiece, unmatchable elements to make the grid searching algorithm for groups of 4 easier, as there would not be bound issues with the search. While I did not ever get to implement the grid search algorithm, the other tips helped encapsulate my game in terms of Java elements, and the insight into grid searching was still nice to learn anyway. I started by creating the Puyo class, realizing early on that it would be a good idea to represent Puyos as objects within the code and taking inspiration from how I developed my card games: first by creating a Card class and then creating a Deck class filled with Cards. At first, the class was simple, with the only instance variable being an integer representing its color and mutator methods for it accordingly. From there, I made a PuyoPair class, a class that represents a PuyoPair, or one Puyo Puyo piece. At first, it had instance variables that held both the top and bottom Puyos in the pair and their board coordinates, but that would soon change, which will be elaborated upon with its respective day. The “board coordinates” of a Puyo are its coordinates within the board 2D array, which I planned on converting into pixel coordinates later. I also made mutator methods for the board coordinates within the PuyoPair class. Finally, I created the Board class, a class that would handle all of the game logic that would be passed to the graphics rendering NullpoPuyo class. I got the idea from a Java Tetris clone tutorial that I also referenced, linked [here](http://zetcode.com/tutorials/javagamestutorial/tetris/). The instance variables were an ArrayList representing the two piece preview window the player is given, a 2D array of ints representing the board, which will change later, a PuyoPair representing the active puyo pair that the player has control over, and a boolean that represents whether or not the player has reached the failstate of the game, or topped out the board. I also wrote some methods for the basics of game logic. There was a method that would fill the piece preview, one that would generate a new PuyoPair, one that would move the next pair from the piece preview to the game board and make it active, mutators for the activePuyoPair and its position, and one that set the entire board to a value, useful for the initialization of the game. Another three hours were down, and I felt like I was making good progress.

On the third day of development, 5/29, I moved to my grandparents’ house so that ours could be fumigated. I did not know at the time that it would last three days, 5/29-5/31; I was hoping it was going to be quick. Being on a setup that was not my own did not make development easier, but nonetheless, I persisted. I realized that I needed to do a rewrite of some of my classes. In hindsight, it is rather obvious that the board coordinates of a Puyo should be stored within the individual Puyo object, not the PuyoPair object, and I rewrote my classes accordingly. I moved everything related to board coordinates within the PuyoPair class into the Puyo class, instance variables and mutator methods included. I also realized that I needed a way to turn the random integers I was generating for Puyo colors into Strings representing the color itself, due to how I was passing in sprite names into my drawSprite method, which followed the format of “PPTPuyo” + (color). This is where I came to Alejandro for guidance; he taught me about enums and how to use them, especially with respect to my project, and a Color enum and lookup method was written. The Board class also needed rewrites, as I also realized that some of the decisions I had made earlier did not make sense with respect to the code I had just written. I rewrote the board 2D array to be of type Puyo instead of int, realized I defined its dimensions wrong and flipped them, created an emptyPuyo to fill and replace the board with, and rewrote my pair and board manipulation methods to work with a 2D array of type Puyo instead of int. From there, I rendered a full board of Puyos and got the pixel coordinates of the corners of the rectangle that would form the board. All Puyos are 64 x 64 pixels, so it was only a matter of developing a formula to convert board coordinates to pixel coordinates, which I did in the rowToPixels and colToPixels methods. Finally, I developed a method that got the lowest empty row within a column, which would be used for my dropping and PuyoPair placement methods within the board. Within the NullpoPuyo class, I instantiated a board and added input handler if statements within the render method in preparation for my future code. I also helped Nick and James with their capstone projects, helping them develop ideas and code alike. Another 2.5 hours had been spent, and while I was slowing down a little, I still felt good about the project.

The fourth day of development, 5/30, was a major breakthrough for me. I was able to render game pieces and make them move based on user input. They did not move correctly or by the rules of the game, but they did move. In order to facilitate this, I wrote the main basic piece movement methods for the game: move left, right, rotate, and drop. I then called them when the input listener in the graphics rendering method received a certain input. Besides that, I wrote more methods that I thought would be useful for game logic, mainly more mutator methods and an empty board check, for “perfect clear” checking, which is when the board is completely cleared after the first piece has been placed. I then added an orientation instance variable for the PuyoPair class, as I determined that knowing a PuyoPair’s orientation would be important for the rotation methods I was writing, which also came with more mutator methods I had to write. I also added a PuyoString class, which was supposed to represent a string of four or more orthogonally adjacent Puyos, but that would not last for long. This day required a lot of thinking, as I had to theorize how to translate inputs into certain movements and handle their edge cases accordingly.

The fifth and final day of development, 5/31, would be the final day of development. I spent a lot of time with my elderly family while I was still at their house, and I could not afford much time to develop the project. I kept the PuyoString class just in case, but I ultimately realized that I was needlessly encapsulating an ArrayList. To replace it, I created an instance variable Puyo ArrayList puyoString within the Board class. I also wrote a method to pop puyoStrings and tried to edit my graphics rendering code to resize itself to fit any display, both of which do not work as far as I am aware. It was a rather uneventful day, but it was a well needed break, I believe.

Overall, while my project did not quite come to fruition as I had hoped it would, it was still a great learning experience for me as a whole. I learned a lot about the time and effort required to develop even the simplest of games, not to mention the great discussion I was able to have with my peers and older, cooler college friend. I hope that I can revisit this project in the future and give it the finish it deserves, or at the very least, give it the barest of functionality. I really enjoyed working on this project, even if it was frustrating at times, and it has provided me a framework for future game development even beyond this one. I know the skills I have been developing even beyond APCSA will help me greatly in the future, and I hope I will be able to continue developing programs, even if they are not necessarily games, as I graduate.