Cory Jbara

Professor Emrich

CSE 20212

**Individual Report**

For our project, Dan Finnegan and I worked on Super Mario Bros, the classic Nintendo platformer for the NES. When the project first began, we talked a lot about the different classes we would need to create the game, and how we would think about problems like objects rendering off screen and collision detection.

My main role in the project was getting Mario on the screen. While this may seem trivial at first, it was actually very difficult to implement. First I got him to render to the screen. Next, I got the right arrow key working, which was not easy. Mario has a ton of minute details in his physical movements, which was hard to replicate in the coding of his actual character movements. He has three different running accelerations, a different skidding deceleration, and very radical aerial physics. All of these were slowly implemented, starting with simply moving right, then adding the left motion, then adding the ability to jump, and finally adjusting his x coordinates in the air (the aerial x physics are actually different than the ground x physics). Also, I had to implement skidding, or else it took Mario way too long to slow down or change directions.

Once I implemented Mario’s physics engine, he was able to move around the screen very effectively. However, the code was not easily reusable, so I had to create the Mario class, with very easy functions to call from the main program. I wrapped the physics into an input handling class. Also, I created a move class and a render class, so that it was very easy to render Mario around the screen.

It was very useful to have this class because once it was completed, we did not have to worry how certain functions would act, we just had to call them.

My other big role in the project was to create the map. I created the NonMoving abstract class, which is basically a general map element that has different hitboxes and ways it can interact with the environment. From this class there are many derived classes, including Brick, Question, Pipe, Stair, and Flag. These also have different ways to interact with the other elements on screen. Once I created the base class, creating the derived classes was rather simple. They all had a render function from the base class and the collision detection with Mario was handled in Mario’s class. All they had to do was return their hitbox, and let the Mario class do most of the work.

Dan did an amazing job with his parts of the projects. He implemented the enemies class and it works very well. He also did a great job on the collision detection with the enemies and the map, as well as Mario and the enemies. We were always on the same page when it came to the tasks we had to accomplish next. We knew where we wanted the project to go and we assigned tasks accordingly. Dan also implemented the mushroom class with the collision detection for that and Mario, and I implemented Mario growing in size once he hit the mushroom. This kind of collaboration was very useful for us, and I believe it made our project amazing. While we should have started coding earlier in the semester in order to create a more elaborate game, it was still very cool to see our game come together piece by piece. When we were in the lab together, which was frequently, we were able to solve some of the biggest problems of the project, such as the collision detection. We worked side-by-side debugging this piece of code for many hours, but in the end we got a very good working product. This is the case for the whole game; I know that without our great collaboration, our game would not have turned out so well.