Project Overview:

In this project we created a game based after the classic Atari game, Pong. In our game there are two user controlled paddles (one on the top of the screen, one on the bottom of the screen) and one bouncing ball. The ball can bounce off the side walls and the paddles. If the ball hits the top or bottom wall a point is awarded to the opposite player.

Results:

We were able to complete our MVP of having just two paddles that moved one unit per click and ball bouncing and more. When we first planned the game we knew we would be taking pre existing code for paddles and bouncing balls and revising it to fit our needs. The paddle code we took moved with the mouse moving so we had a lot of work ahead of us. We were able to get the paddles to move by pushing keys and were happy leaving it like that. However, later we wanted to stretch a little further so we found out how to make the paddles move by pressing and holding a key. The ball bounce code we used gravity and physics, which we didn't really need. We simply needed the ball to bounce of walls at prescribed angles (and then know the difference between a wall and a paddle for the top and bottom walls). With some work we were able to get a great MVP working.

We then went beyond by tracking score (game play shown in second image, including the updating scores). We recorded the score in terminal giving each player a point when the ball hit the wall opposite to the paddle. We didn't want one continuous game so we then figured out how to make the game restart when the score got to a certain value. Playing like this was fun because you could have a game ending! But we wanted to be able to know when the game restarted so we found a way to have "Player _ Wins" once that player reached the winning score before the game reset (this screen the third picture below). We also wanted the entrance to the game to be more visually pleasing so we worked to be able to have a Welcome page that tells rules and doesn't start the game until the spacebar is pressed (shown below in first image). Now, we have a final product that we are very pleased with.

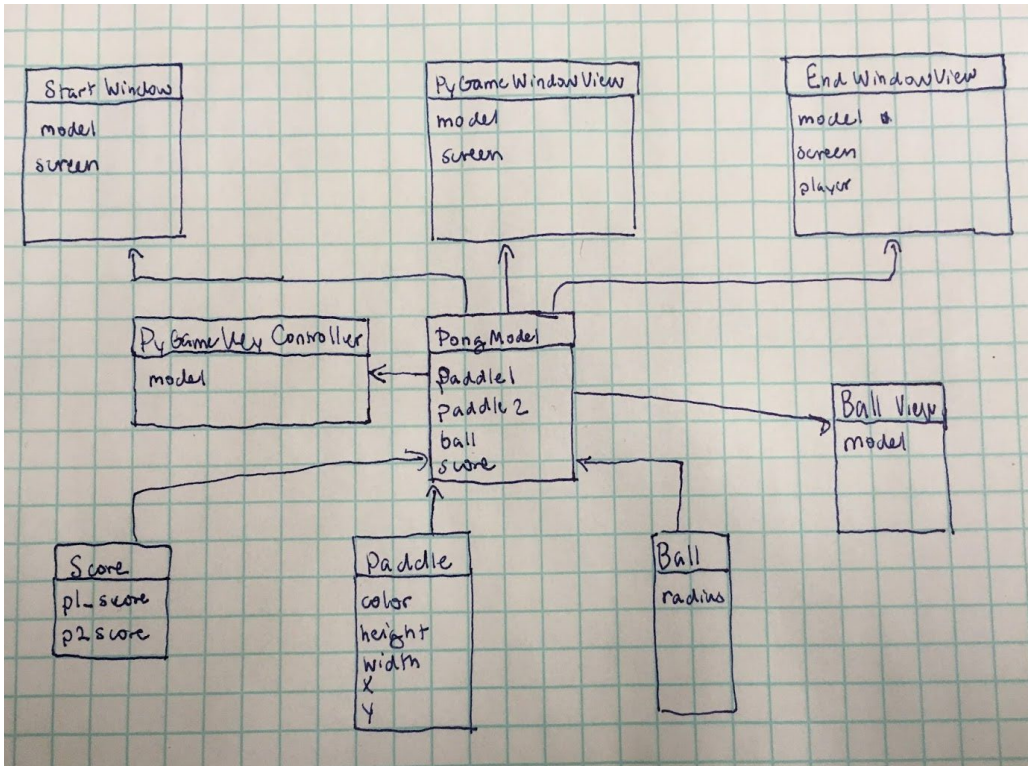

Pro Pong
Welcome to Pro Pong
Player1: (bottom) arrow keys
Player2: (top) A move left, D move right
Press Space Bar to Start


Pro Pong
PLAYER 2 SCORE: 0
PLAYER 1 SCORE: 2

Implementation:

Our code has classes for both displaying objects and keeping track of their attributes. In general, an object's attributes were kept in one class, the object was drawn in another class and then all the draw classes were called in a separate display class. For example, the position, speed, and direction of the ball is kept track of in the Ball class. However, the ball is drawn in the BallView class and called in the PyGameWindow class. This makes it so it easier to manipulate the numerical components of the ball and the visual. This same logic and organization was used for the score and paddles as well.

An important design decision we made was having three different views of the game instead of changing dramatically what was displayed on one view. When the game is opened there is a simple welcome page with instructions. The most complicated view is the game view and there is also a view of the game when a player wins. Each of these views has objects associated with it and can be called on or off.



Reflection:

Overall, our project went really well. We started a little late because we were both preoccupied with other things. However, when we hunkered down and worked we were able to get a lot done. We were able to divide work fairly well. At some times it felt one partner was doing a little more work but we talked and were able to work past any qualms that might have been. Though we didn't devote many days to the project, we devoted a lot of hours on the days that we worked. Perhaps next time we would want to spread our work out more, but this worked really well for us. We were able to get a lot done and feel accomplished with what we completed. We both succeeded in our learning goals of learning more about GUI and interactive programming. We both feel we have a much better sense on the topics, so we could use them again in a different context. We communicated well as a pair and I think that is what really helped us succeed.

Initially, we discussed doing all the coding at the same time. However, with our different schedules we discovered it was easier to assign each other concrete tasks to accomplish on our own time. When we were having trouble we came to each other and were often able to solve the problem together. The splitting of specific tasks also helped with version control in Github because we were not changing the same aspects of the code.