Sara Wescott

CS 172-2

Project: Exploding Kittens

Partners: Oscar, Jacob

In the project, Exploding Kittens, I helped create the UML after playing the game with my group. The two classes we made were the Player class and the card class. All players can hold cards, so first thing was to make the card class. I was assigned to complete the card class which was only the public function getType and the private variable string type (which was the type of card).

Once completed, my partners and I began working on the Player class. My four functions were the constructor, SeeTheFuture, LookUp, and Shuffle. The constructor was a custom constructor that allowed a person to put in a name for the player, and then added 5 cards into their hand. One of these cards was the defuse card and the others were random. The SeeTheFuture function allowed the player that played that card to look at the first three cards at the top of the deck. I did add a “no peeking” line for the other players so that they would not cheat. The LookUp function reads out the instructions of a card that the player puts in from their hand. This helps the player know how to play it and so on. The Shuffle function is a card that can be played, but was later changed. When a person plays the shuffle card, it shuffles the deck, but because the deck is not a part of the Player function, it had no meaning to the class. It was later implemented as a template in function.h for easy access. I later added the getDead bool function which was not in the original UML.

Once all the functions were compiled, the entire team worked out all the bugs in the code. Then we began working on the main code. In the beginning, I went through Jacob’s main code which had the deck and player creation with the rest of the team. The problems fixed included constructor errors and Player functions that did not match up with the player creation. This was the point where the shuffle function had to be changed. After this, we worked on the turn base loops for the main game. The loops called for a player to play a card. Depending on the card, different if and if else statements were made. The cards I coded were favor and pair. Pair called for a linear search to check that there were two cards, which I added. Pair then deleted the other card that matched, or if it did not match it would start all over asking the person what card they wanted to played.

As the last day we worked on the project went, I ended up working a large number of functions and issues with the main. Many bugs came from each team member adding together their own versions of functions. Which made it difficult because we had to understand their code and then compile it together. We had to do this several times which made the project difficult, but we did it often in order to avoid missing something. By the end of this I worked on functions inside the main and in the player class, these included tweaking favor, pair, skip, addCard – a new function of player class for favor and pair, and removeHand – another new function for the player class. AddCard was added because we needed a way to and the card that was chosen from pair and favor to the players hand. Then removeHand was written to remove a card from a players hand. Which was not simple because the hand class did not like the erase function so I wrote my own “erase.” I put cards into a dummy hand except the one that needed to be taken out and then cleared the hand. I put the dummy hand cards back into the hand and everything was fixed.