The System Development Life Cycle (SDLC)

For this project, the incremental SDLC model was used, where the basic functionality of the game was completed early and most difficult milestones, such as the integration of all the classes was implemented first. Five cycles were made in total, and the following milestones were completed with each cycle:

Cycle 1

The first group meeting was held, the initial planning was made and each teammate was assigned a task (to create one or two classes of the four classes). After creating the four classes, which are, the card, deck, player and game class, with their appropriate constructors and basic methods (such as the choose-suit method in the player class), another group meeting was held, where the classes were integrated and tested.

Cycle 2

Further planning was made and each teammate was assigned a task to further develop the classes. The shuffling functionality was implemented in the deck class, the deal method, draw-card and play-game methods were then created in the game class. In the play class, a play method was created to allow for several modes of play. Testing of these new methods was made whilst they were being created and backed up the end. At this point in the project, several human player objects could be created, dealt with cards, and could play cards onto the discard pile, however, the players could play invalid moves because no validation process had been implemented.

Cycle 3

Another meeting was held, and other tasks were assigned to teammates. The check-valid method, which validated the players’ moves was implemented and another play method in the player class (that allows players to defend themselves in an attack situation) was created. Along with this method, a method that reads the abilities of the cards was created to validate cards that have special abilities. Finally, the current progress was documented, the code was tested and backed up.

Cycle 4

The current classes were completed and a new bot class, for the computer player was created. This class extended the player class and used polymorphism to allow the computer and human players to be in a single array. Finally, a GUI class was created. However, due to time constraints, it was not integrated with the final game, and more important aspects were focused upon. In future, though, it is possible to link the GUI to the game, for a more visually appealing front-end. Nonetheless, the text-based interface was made to look neat and user friendly.

Cycle 5

The final documentation and testing of the code were made. The code and documentation were then uploaded to GitHub.