Deliverable 1

Description:

The goal of this project is to recreate the card game “War” with two players in mind via Java.

How to play the game:

With two players, the deck is evenly split between both players. 26 random cards for player 1 and 26 for player 2. On execution, the game will prompt player 1 and player 2 to enter their usernames. After that, the game will automatically run and compare the value of player 1’s card and player 2’s card and determine which player won, lost, or if it was a draw. The player with the greater card value wins the round, “2” being the lowest value and “Ace” being the highest value. This sequence will then run until all cards are exhausted and the total amount of rounds won at the end will determine the victor.

The base code of the game:

Using Java, the code of our game primarily consists of three groups of code. One for the player's name entry, one for our cards and their values, and one for ending the game and tallying up the scores. Before we added any classes, we established both a scanner to allow our users to enter their name, and we established an array list to keep track of card counts and their values. In our first code group, we simply used two methods for the player names and just set a scanner to allow the user to type in their names each. From there, it will take those names and store them in two name integers used for the end of the game. Our second code group is the most important one as it keeps track of all the cards, and their values, which are randomized using an array and a for loop for each player. The final code group simply tallies up the final score and displays who the winner is.

Scope of our project:

Jonah will handle the base code, the general structure of the classes, and main. Mustafa will create the UML of the game as well as handle code testing and debugging. Dylan will write up the design document and track task progression. Ali will handle additional code testing and debugging.

When the game's code is functionally able to take user input for the names of player 1 and 2, compare values against each other and determine the greater value and assign the point to the winner. Then run 26 rounds of card comparisons or until all cards are exhausted and then total all points to determine a winner. Lastly, a UML that logs and displays the classes and methods used to create the game. These aspects will dictate a complete project.

Implementation Plan:

Repository URL:

https://github.com/JonathanJoestarr/Deliverable-1

Each group member will look over each other's work and send feedback. If a member decides to make potential changes, they will be done via branches. Afterward, the original creator will have the final say on merging. Base code and its branches will be within their own folder, UMLs and other visual representations and diagrams will be in their own folder, and documents will be in their own folder, keeping all three aspects organized.

The game program will be developed via NetBeans and the UML will be made using Visual Paradigm. The coding itself will focus on readability for all developers with no inside jargon. Classes will also be separated into their own files so all developers can see at a glance what files contain what classes at all times.

Design considerations:

When it comes to the design of our game, we primarily used for loops, if statements, and arrays, in tangent with some object-oriented protocols. Our for loops were used to automate and randomize the value of the cards that are drawn by both players. We used if statements to both determine who wins the round, and the game. Our array list was used to keep track of the deck size, and the values of the cards. In terms of our OOP protocols, we used constructors to keep track of our player names, and the final tally.

UML Class Diagram