Aleksandar Knezevic

Caglar Kurtkaya

Emily Lin

Dayana Roa-Tapia

Section: 12:30pm

**3’s**

Team 1

**Section 1:**

3’s is a multiple player die game where the player with the least points wins.

It is played with 5 dice. Each player rolls all 5 dice at the same time. Each player must take at least one dice each turn but can take more than one. The number show on the dice taken will be added to the total points of the player. After the player takes the dice that they want, they must roll the remaining dice and repeat the process of taking one or more dice until there are no more dice to roll.

If the number shown on a dice is a 3, then that dice is worth zero points. This means that the lowest number of points a player can get is zero, given that the player rolls a 3 on every single dice by the end of their turn.

The goal of each player is to beat the player with the least amount of points accumulated.

**Section 2:**

The game will have a client and server implementation which takes commands as events from the GUI.

The server will be in charge of keeping track of how many players (4 in this case) are in the game, who the winning player is, and of who wants to play again. If not all players want to play again the server will let the other players that they are waiting on players to connect. It is also in charge of starting and ending the game. Throughout the game, the server will allow all players to see the die and what each player gets and takes on each round. The score to beat along with the player name will be displayed on each of the player’s screen, which is being sent to the client by the server. The player’s own score will also be displayed for each player.

The client will have the option to roll the die and to select which dice they want to take from each turn. Each roll and points will be sent from client to server so that the server can display it on the GUI for all the players in the game.

When all the players have had a turn, the server will check to see who has the lowest score and will send the player name and score to all the players and display a winning message that includes the player and score.

**Section 3:**

Using Node.js and the HTTP module in Node.js we will create the server and client connections along with creating the GUI.

For the GUI on the server side we will create a text box that shows the players connected or if waiting for player to connect. It will also show the players and how many points each player currently has.

The client side of the GUI will display the score and player to beat, the player’s own score, and the die being rolled. Each client will be able to see the die being rolled and taken even if it is not their turn. The client will also have a button to roll the die and one to quit. If a player decides to quit in the middle of the game then that client will disappear but the game will continue.

Once every player has had their turn, the winning message will display on the screen of all players. The winning screen should have the option to quit or to play again.

**Section 4:**

Benefits:

1. The players don’t have to install a specific IDE in order to run the game.
2. The server will allow all players to see the die and what each player rolls and takes on each round. In this case, all players know each player’s status
3. The game is easy to understand and play
4. The reason why we chose to implement the game in Node.js is because it’s much easier to implement than Java. In Java, the server and the client are separated. In Node.js, we can combine the server and the client into one file and use an event handler to handle the connection

Issues/Risk:

1. Since it’s a 4 player game, the game will not start until there’s 4 players connected to the server. So if there’s only 3 players connected to the server, the game will never start