PROMINEO TECH

Intro to JavaScript Week 6 Coding Assignment

Points possible: 100

URL to GitHub Repository:

https://github.com/FSWEric/FSWEric-JS-Week6-Coding-Assignment/upload/main

URL to Your Coding Assignment Video:

https://youtu.be/K_cE3jkiYi4

Instructions: In Visual Studio Code, or an IDE of your choice, write the code that accomplishes the objectives listed below. Ensure that the code compiles and runs as directed. Take screenshots of the code and of the running program (make sure to get screenshots of all required functionality) and paste them in this document where instructed below. Create a new repository on GitHub for this week's assignments and push this document, with your JavaScript project code, to the repository. Add the URL for this week's repository to this document where instructed and submit this document to your instructor when complete.

Coding Steps:

For the final project you will be creating an automated version of the classic card game *WAR*. You do not need to accept any user input, when you run your code, the entire game should play out instantly without any user input.

There are many versions of the game *WAR,* but in this version there are only 2 players and you don't need to do anything special when there is a tie on a round.

Think about how you would build this project and write your plan down. Consider classes such as Card, Deck, and Player and what fields and methods they might each have. You can implement the game however you'd like (i.e. printing to the console, using alert, or some other way). The completed project should, when run, do the following:

- Deal 26 Cards to two Players from a Deck.
- Iterate through the turns where each Player plays a Card
- The Player who played the higher card is awarded a point
    - Ties result in zero points for both Players

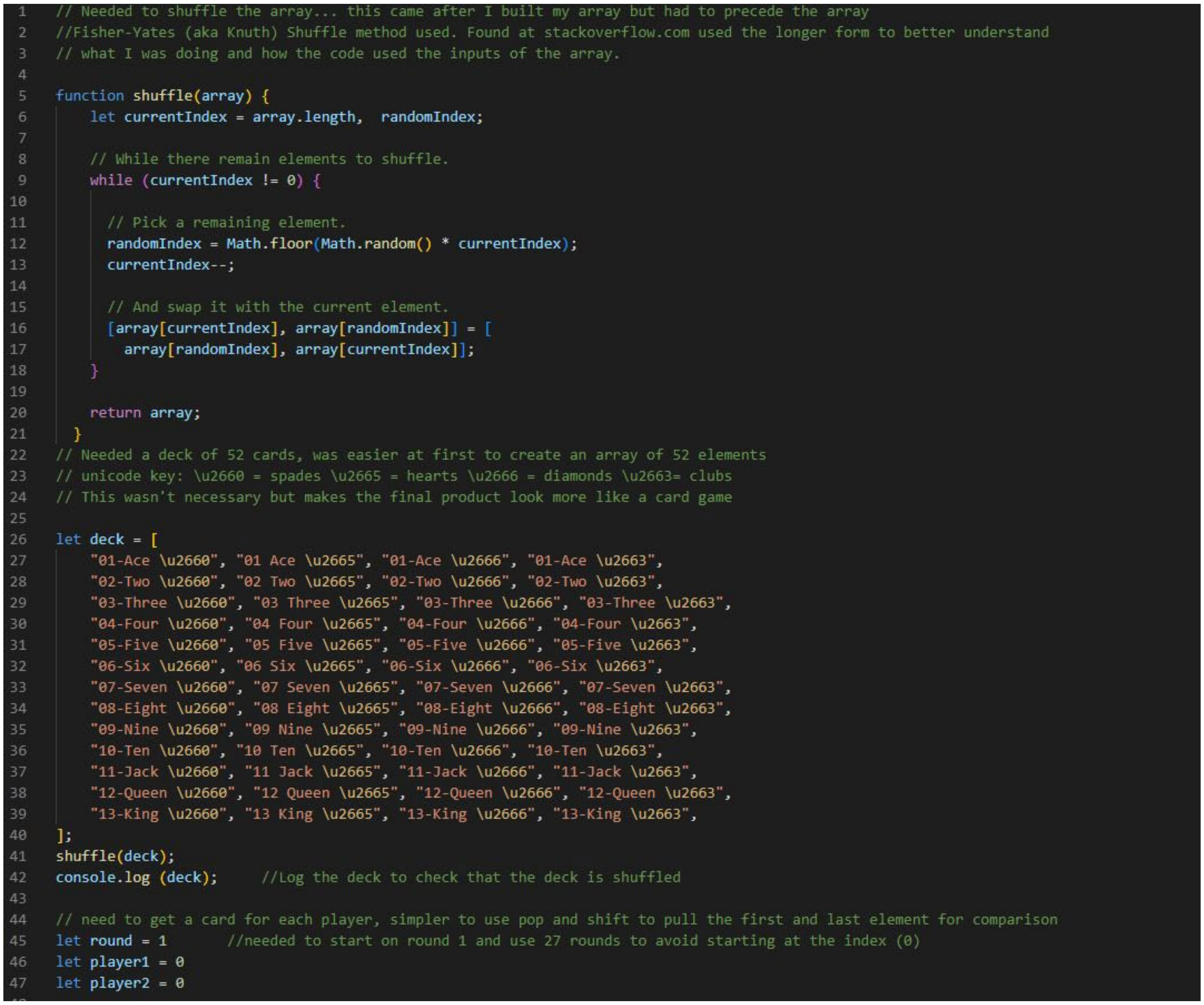
- After all cards have been played, display the score and declare the winner.

Write a Unit Test using Mocha and Chai for at least one of the functions you write.

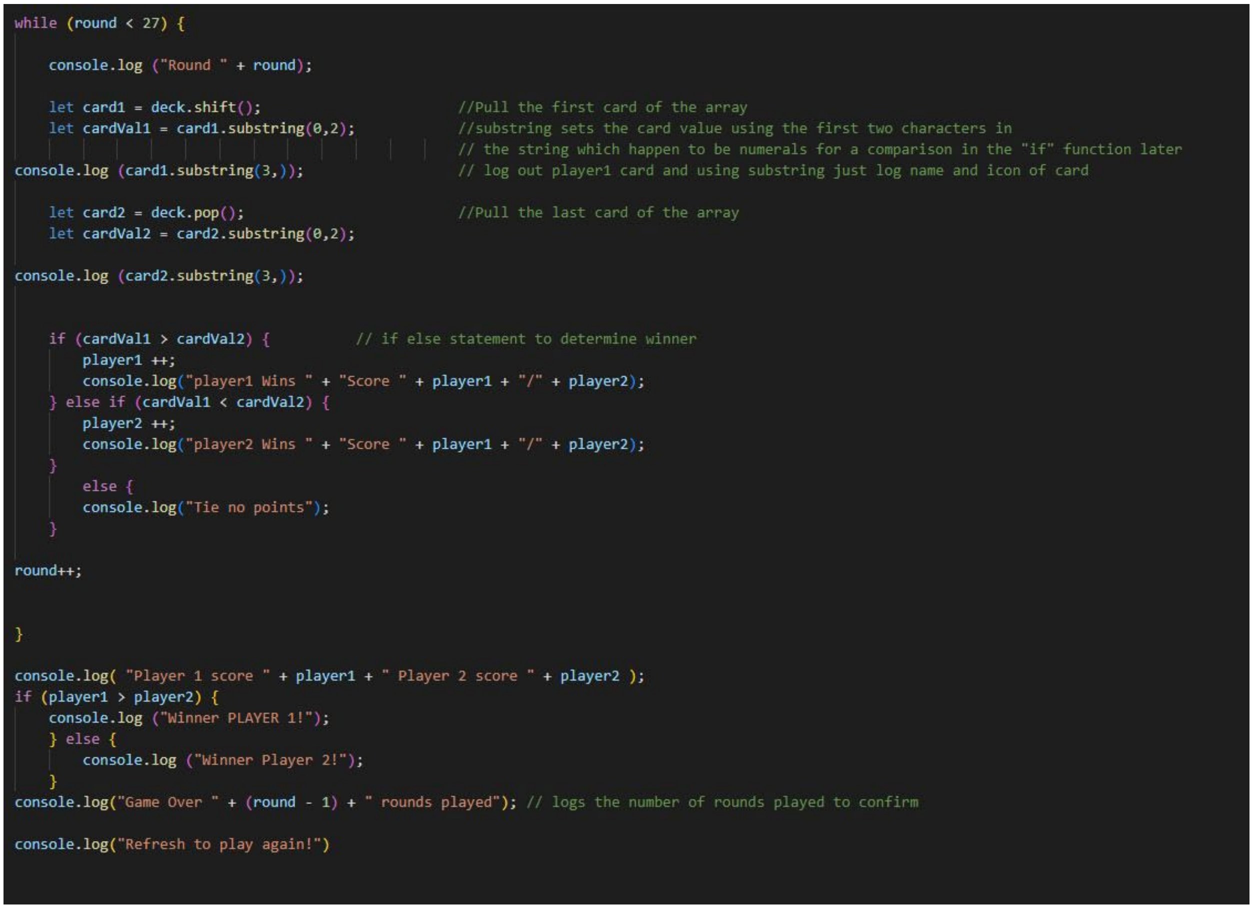
Video Steps:

Create a video, up to five minutes max, showing and explaining how your project works with an emphasis on the portions you contributed. This video should be done using screen share and voice over. This can easily be done using Zoom, although you don't have to use Zoom, it's just what we recommend. You can create a new meeting, start screen sharing, and start recording. This will create a video recording on your computer. This should then be uploaded to a publicly accessible site, such as YouTube, Dropbox, or Google Drive. MAKE SURE THE LINK YOU SHARE IS PUBLIC or UNLISTED. If it is not accessible by your grader, your project will be graded based on what they can access. The link should be pasted in the submission text box after the GitHub repo link. REQUIRED: PUBLIC link to video, and GitHub repo link with everything listed above!

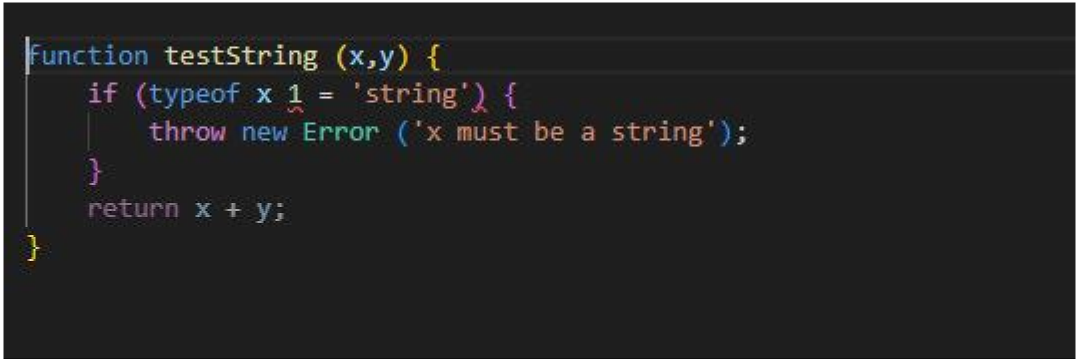
Screenshots of Code:



```javascript
// Needed to shuffle the array... this came after I built my array but had to precede the array
//Fisher-Yates (aka Knuth) Shuffle method used. Found at stackoverflow.com used the longer form to better understand
// what I was doing and how the code used the inputs of the array.

function shuffle(array) {
    let currentIndex = array.length,  randomIndex;

    // While there remain elements to shuffle.
    while (currentIndex != 0) {

      // Pick a remaining element.
      randomIndex = Math.floor(Math.random() * currentIndex);
      currentIndex--;

      // And swap it with the current element.
      [array[currentIndex], array[randomIndex]] = [
        array[randomIndex], array[currentIndex]];
    }

    return array;
  }
// Needed a deck of 52 cards, was easier at first to create an array of 52 elements
// unicode key: \u2660 = spades \u2665 = hearts \u2666 = diamonds \u2663= clubs
// This wasn't necessary but makes the final product look more like a card game

let deck = [
    "01-Ace \u2660", "01 Ace \u2665", "01-Ace \u2666", "01-Ace \u2663",
    "02-Two \u2660", "02 Two \u2665", "02-Two \u2666", "02-Two \u2663",
    "03-Three \u2660", "03 Three \u2665", "03-Three \u2666", "03-Three \u2663",
    "04-Four \u2660", "04 Four \u2665", "04-Four \u2666", "04-Four \u2663",
    "05-Five \u2660", "05 Five \u2665", "05-Five \u2666", "05-Five \u2663",
    "06-Six \u2660", "06 Six \u2665", "06-Six \u2666", "06-Six \u2663",
    "07-Seven \u2660", "07 Seven \u2665", "07-Seven \u2666", "07-Seven \u2663",
    "08-Eight \u2660", "08 Eight \u2665", "08-Eight \u2666", "08-Eight \u2663",
    "09-Nine \u2660", "09 Nine \u2665", "09-Nine \u2666", "09-Nine \u2663",
    "10-Ten \u2660", "10 Ten \u2665", "10-Ten \u2666", "10-Ten \u2663",
    "11-Jack \u2660", "11 Jack \u2665", "11-Jack \u2666", "11-Jack \u2663",
    "12-Queen \u2660", "12 Queen \u2665", "12-Queen \u2666", "12-Queen \u2663",
    "13-King \u2660", "13 King \u2665", "13-King \u2666", "13-King \u2663",
];
shuffle(deck);
console.log (deck);     //Log the deck to check that the deck is shuffled

// need to get a card for each player, simpler to use pop and shift to pull the first and last element for comparison
let round = 1       //needed to start on round 1 and use 27 rounds to avoid starting at the index (0)
let player1 = 0
let player2 = 0
```

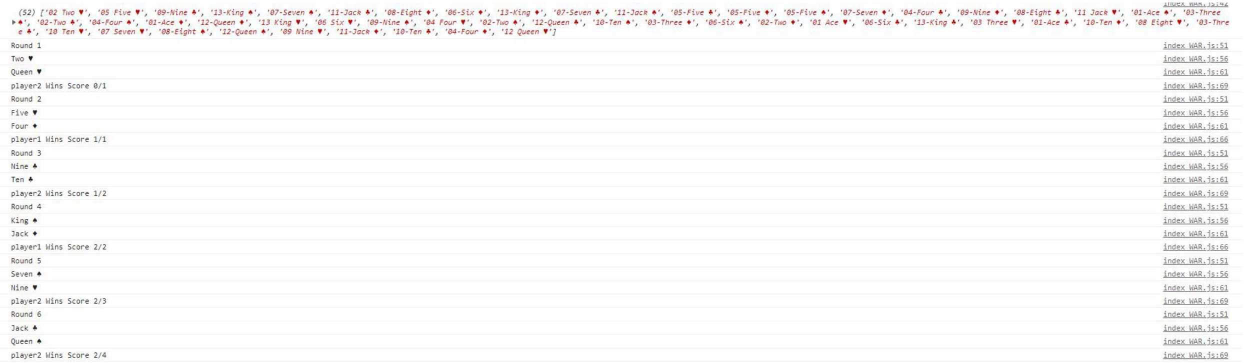


```
while (round < 27) {

    console.log ("Round " + round);

    let card1 = deck.shift();                        //Pull the first card of the array
    let cardVal1 = card1.substring(0,2);             //substring sets the card value using the first two characters in
                                                     // the string which happen to be numerals for a comparison in the "if" function later
console.log (card1.substring(3,));                   // log out player1 card and using substring just log name and icon of card

    let card2 = deck.pop();                          //Pull the last card of the array
    let cardVal2 = card2.substring(0,2);

console.log (card2.substring(3,));


    if (cardVal1 > cardVal2) {           // if else statement to determine winner
        player1 ++;
        console.log("player1 Wins " + "Score " + player1 + "/" + player2);
    } else if (cardVal1 < cardVal2) {
        player2 ++;
        console.log("player2 Wins " + "Score " + player1 + "/" + player2);
    }
        else {
        console.log("Tie no points");
    }

round++;


}

console.log( "Player 1 score " + player1 + " Player 2 score " + player2 );
if (player1 > player2) {
    console.log ("Winner PLAYER 1!");
    } else {
        console.log ("Winner Player 2!");
    }
console.log("Game Over " + (round - 1) + " rounds played"); // logs the number of rounds played to confirm

console.log("Refresh to play again!")
```

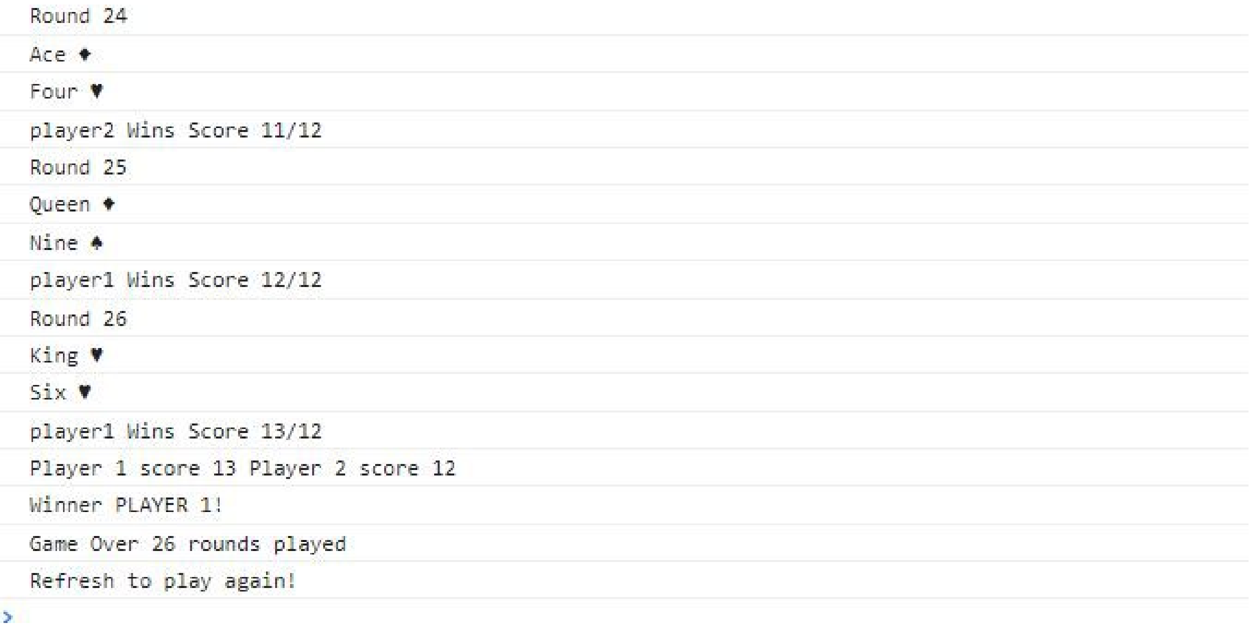


```
function testString (x,y) {
    if (typeof x 1 = 'string') {
        throw new Error ('x must be a string');
    }
    return x + y;
}
```

Screenshots of Running Application:



```
                                                                                                             index WAR.js:42
  (52) ['02 Two ♥', '05 Five ♥', '09-Nine ♣', '13-King ♠', '07-Seven ♠', '11-Jack ♣', '08-Eight ♦', '06-Six ♦', '13-King ♦', '07-Seven ♣', '11-Jack ♠', '05-Five ♣', '05-Five ♦', '05-Five ♠', '07-Seven ♦', '04-Four ♣', '09-Nine ♦', '08-Eight ♣', '11 Jack ♥', '01-Ace ♠', '03-Three
▸♠', '02-Two ♣', '04-Four ♠', '01-Ace ♦', '12-Queen ♦', '13 King ♥', '06 Six ♥', '09-Nine ♠', '04 Four ♥', '02-Two ♠', '12-Queen ♣', '10-Ten ♠', '03-Three ♦', '06-Six ♠', '02-Two ♦', '01 Ace ♥', '06-Six ♣', '13-King ♣', '03 Three ♥', '01-Ace ♣', '10-Ten ♦', '08 Eight ♥', '03-Thre
  e ♣', '10 Ten ♥', '07 Seven ♥', '08-Eight ♠', '12-Queen ♠', '09 Nine ♥', '11-Jack ♦', '10-Ten ♣', '04-Four ♦', '12 Queen ♥']
Round 1                                                                                                      index WAR.js:51
Two ♥                                                                                                        index WAR.js:56
Queen ♥                                                                                                      index WAR.js:61
player2 Wins Score 0/1                                                                                       index WAR.js:69
Round 2                                                                                                      index WAR.js:51
Five ♥                                                                                                       index WAR.js:56
Four ♦                                                                                                       index WAR.js:61
player1 Wins Score 1/1                                                                                       index WAR.js:66
Round 3                                                                                                      index WAR.js:51
Nine ♣                                                                                                       index WAR.js:56
Ten ♣                                                                                                        index WAR.js:61
player2 Wins Score 1/2                                                                                       index WAR.js:69
Round 4                                                                                                      index WAR.js:51
King ♠                                                                                                       index WAR.js:56
Jack ♦                                                                                                       index WAR.js:61
player1 Wins Score 2/2                                                                                       index WAR.js:66
Round 5                                                                                                      index WAR.js:51
Seven ♠                                                                                                      index WAR.js:56
Nine ♥                                                                                                       index WAR.js:61
player2 Wins Score 2/3                                                                                       index WAR.js:69
Round 6                                                                                                      index WAR.js:51
Jack ♣                                                                                                       index WAR.js:56
Queen ♠                                                                                                      index WAR.js:61
player2 Wins Score 2/4                                                                                       index WAR.js:69
```



```
Round 24
Ace ♣
Four ♥
player2 Wins Score 11/12
Round 25
Queen ♦
Nine ♠
player1 Wins Score 12/12
Round 26
King ♥
Six ♥
player1 Wins Score 13/12
Player 1 score 13 Player 2 score 12
Winner PLAYER 1!
Game Over 26 rounds played
Refresh to play again!
>
```