Week 6: Final Project

URL to GitHub Repository: https://github.com/nmason90/Week-6-Coding-Assignment.git

URL to Your Coding Assignment Video: https://youtu.be/qkOS_aARDs8

Instructions:

- In Visual Studio Code, write the code that accomplishes the objectives listed below and ensures that the code compiles and runs as directed.
- Create a new repository on GitHub for this week's assignments and push this document, with your project code, to the repository.
- Include the URLs for this week's repository and video where instructed.
- Submit this document as a .PDF file in the LMS.

Coding Steps:

- For the final project you will be creating an automated version of the classic card game *WAR!* There are many versions of the game *WAR.* In this version there are only 2 players.
    - You do not need to do anything special when there is a tie in a round.
- Think about how you would build this project and write your plan down. Consider classes such as: **Card**, **Deck**, **Player**, as well as what **properties** and **methods** they may include.
    - 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 inside of your browser's console.

The completed project should, when executed, do the following:

- Deal 26 Cards to each Player from a Deck of 52 cards.
- 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.

PROMINEO TECH

Week 6: Final Project

```
const SUITS = ["Spade", "Heart", "Club", "Diamond"]
const VALUES = ["A", "2", "3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K"]
const CARD_VALUE_MAP = {
    'A': 1,
    '2': 2,
    '3': 3,
    '4': 4,
    '5': 5,
    '6': 6,
    '7': 7,
    '8': 8,
    '9': 9,
    '10': 10,
    'J': 11,
    'Q': 12,
    'K': 13,
} // assigns numerical value to each card in the Deck to be able to compare values, in particular for letter cards,
// and determine which card has the higher value


class Player {
    constructor(name) {
        this.name = name;
        this.score = 0;
        this.hand = [];
    }
    playCard() {
        return this.hand.pop ()
    }
    incrementScore() {
        this.score += 1
    }
} // end Player; creates Player class to generate Player information including name, score (which is incremented with this.score +=1,
  //and player's dealt cards which is held within the array)

class Deck {
    constructor (cards = freshDeck()) {
        this.cards = cards
    }
```

Week 6: Final Project

```
    shuffleDeck () {
        this.cards = this.cards.sort(() => Math.random() - 0.5)
    }
    dealCards (p1, p2) {
        while (this.cards.length > 0) {
            p1.hand.push(this.cards.pop());
            p2.hand.push(this.cards.pop());
        }
    }
} //end Deck; creates Deck class which includes all cards which are generated through the freshDeck function,
  // and shuffles cards which are then dealt to each player

class Card {
    constructor (suit, value) {
        this.suit = suit
        this.value = value
    }
    describe () {
        return `${this.value} of ${this.suit}`
    }
} //end Card; creates Card class by generating card suit and value elements

function freshDeck () {
    return SUITS.flatMap(suit => {
        return VALUES.map(value => {
            return new Card (suit, value)
    })
   })
} // function condenses "suit" and "value" arrays (const listed above) and condenses them into one array to create all cards in
  // a deck


const playerOne = new Player ("Tiger"); // creates new Player
const playerTwo = new Player ("Vivi"); // creates new Player
const gameDeck = new Deck (); // creates Game Deck

gameDeck.shuffleDeck(); // shuffles Deck
gameDeck.dealCards (playerOne, playerTwo); // deals 26 cards into array for both players
console.log(playerOne, playerTwo); // prints out cards in each player's hand
```

Week 6: Final Project

```
80  function comparePoints (playerOne, playerTwo) {
81      for (let i = 0; i < playerOne.hand.length; i++) {
82          if (CARD_VALUE_MAP[playerOne.hand[i].value] > CARD_VALUE_MAP[playerTwo.hand[i].value]) {
83              playerOne.score += 1;
84              console.log(playerOne.hand[i], playerTwo.hand[i]);
85              console.log(`${playerOne.name} is this round's winner!`);
86          } else if (CARD_VALUE_MAP[playerTwo.hand[i].value] > CARD_VALUE_MAP[playerOne.hand[i].value]) {
87              playerTwo.score += 1;
88              console.log(playerOne.hand[i], playerTwo.hand[i])
89              console.log(`${playerTwo.name} is this round's winner!`);
90          } else {
91              console.log(playerOne.hand[i], playerTwo.hand[i])
92              console.log(`It's a tie...no points for ${playerOne.name} or ${playerTwo.name}.`)
93          }
94      }
95  } //creates function to compare points, given "CARD_VALUE_MAP" that assigns numerical values to all cards, and determine
96  //which player has the higher point value each time players' cards are played
97
98  function finalScore (playerOne, playerTwo) {
99      if (playerOne.score > playerTwo.score) {
100         console.log (`${playerOne.name} is the overall winner with a score of ${playerOne.score}!`);
101     } else if (playerTwo.score > playerOne.score) {
102         console.log (`${playerTwo.name} is the overall winner with a score of ${playerTwo.score}!`);
103     } else {
104         console.log(`${playerOne.name} and ${playerTwo.name} have a tie score of ${playerOne.score} - play again to find a winner!`);
105     }
106 } // compares overall score, which is incremented over each iteration of the comparePoints function, to determine the overall
107 //winner of the game (26 iterations are expected before determining a final player)
108
109 comparePoints (playerOne, playerTwo); // prints out each round that compares point values of players' cards
110
111 finalScore (playerOne, playerTwo); // prints out overall score for winning player
112
```

```
▸Player {name: 'Tiger', score: 0, hand: Array(26)} ▸Player {name: 'Vivi', score: 0, hand: Array(26)}                wargame.js:78
▸Card {suit: 'Club', value: 'K'} ▸Card {suit: 'Spade', value: 'A'}                                                  wargame.js:84
Tiger is this round's winner!                                                                                       wargame.js:85
▸Card {suit: 'Heart', value: '2'} ▸Card {suit: 'Diamond', value: '6'}                                               wargame.js:88
Vivi is this round's winner!                                                                                        wargame.js:89
▸Card {suit: 'Spade', value: '7'} ▸Card {suit: 'Club', value: 'Q'}                                                  wargame.js:88
Vivi is this round's winner!                                                                                        wargame.js:89
▸Card {suit: 'Club', value: 'A'} ▸Card {suit: 'Diamond', value: 'J'}                                                wargame.js:88
Vivi is this round's winner!                                                                                        wargame.js:89
▸Card {suit: 'Diamond', value: '3'} ▸Card {suit: 'Club', value: '4'}                                                wargame.js:88
Vivi is this round's winner!                                                                                        wargame.js:89
▸Card {suit: 'Heart', value: '4'} ▸Card {suit: 'Club', value: '10'}                                                 wargame.js:88
Vivi is this round's winner!                                                                                        wargame.js:89
▸Card {suit: 'Diamond', value: '8'} ▸Card {suit: 'Spade', value: 'Q'}                                               wargame.js:88
Vivi is this round's winner!                                                                                        wargame.js:89
▸Card {suit: 'Club', value: 'J'} ▸Card {suit: 'Heart', value: 'Q'}                                                  wargame.js:88
Vivi is this round's winner!                                                                                        wargame.js:89
▸Card {suit: 'Club', value: '5'} ▸Card {suit: 'Heart', value: '8'}                                                  wargame.js:88
Vivi is this round's winner!                                                                                        wargame.js:89
▸Card {suit: 'Spade', value: '8'} ▸Card {suit: 'Spade', value: '9'}                                                 wargame.js:88
Vivi is this round's winner!                                                                                        wargame.js:89
▸Card {suit: 'Club', value: '9'} ▸Card {suit: 'Diamond', value: '10'}                                               wargame.js:88
Vivi is this round's winner!                                                                                        wargame.js:89
▸Card {suit: 'Spade', value: '3'} ▸Card {suit: 'Spade', value: 'J'}                                                 wargame.js:88
Vivi is this round's winner!                                                                                        wargame.js:89
▸Card {suit: 'Club', value: '6'} ▸Card {suit: 'Spade', value: '6'}                                                  wargame.js:91
It's a tie...no points for Tiger or Vivi.                                                                           wargame.js:92
▸Card {suit: 'Heart', value: 'K'} ▸Card {suit: 'Diamond', value: '7'}                                               wargame.js:84
Tiger is this round's winner!                                                                                       wargame.js:85
```

Week 6: Final Project

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.
    - Ensure 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.