PROMINEO TECH

Intro to JavaScript Week 6 Coding Assignment

Points possible: 70

Category	Criteria	% of Grade
Functionality	Does the code work?	25
Organization	Is the code clean and organized? Proper use of white space, syntax, and consistency are utilized. Names and comments are concise and clear.	25
Creativity	Student solved the problems presented in the assignment using creativity and out of the box thinking.	25
Completeness	All requirements of the assignment are complete.	25

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.

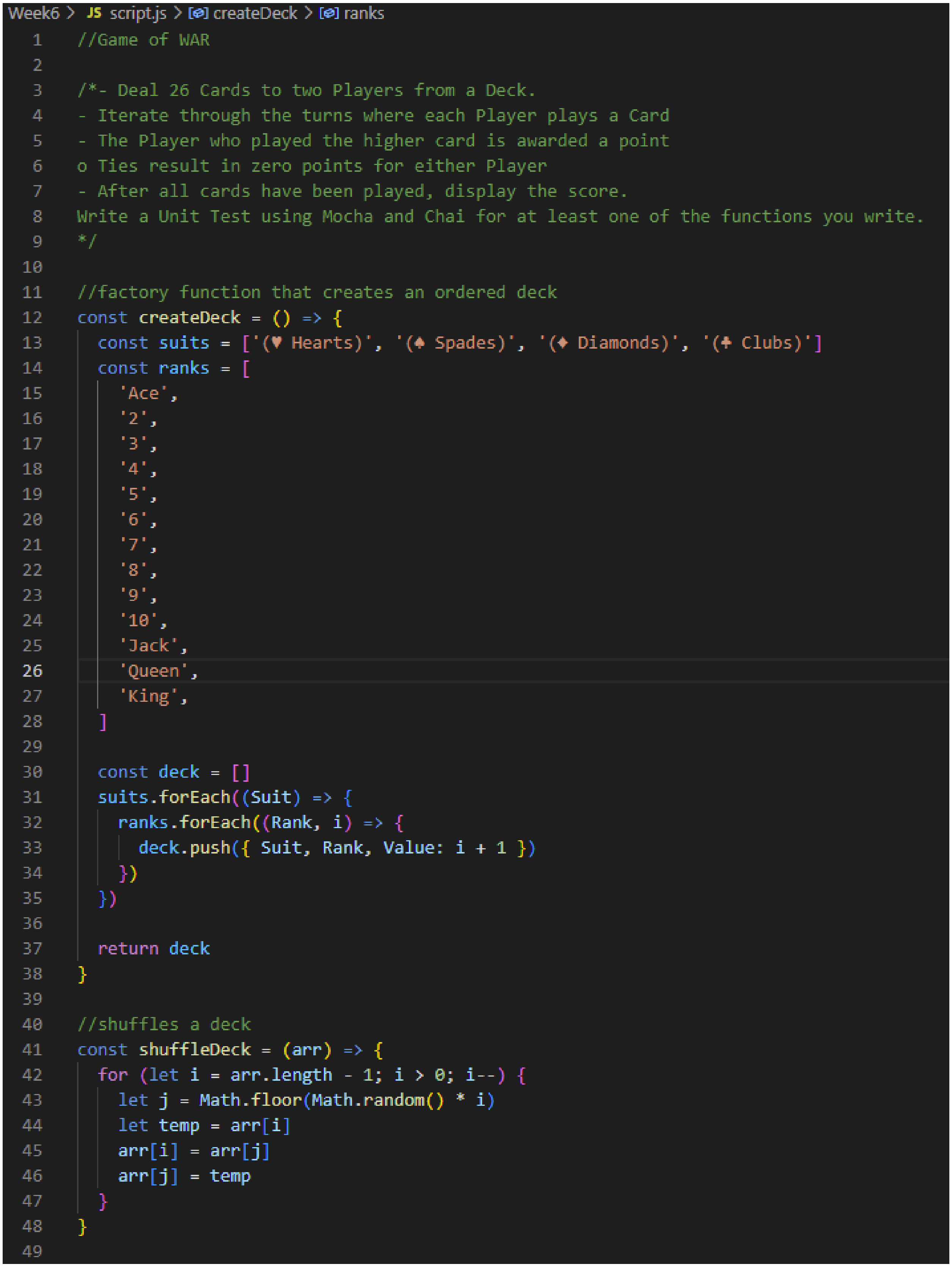
Coding Steps:

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.

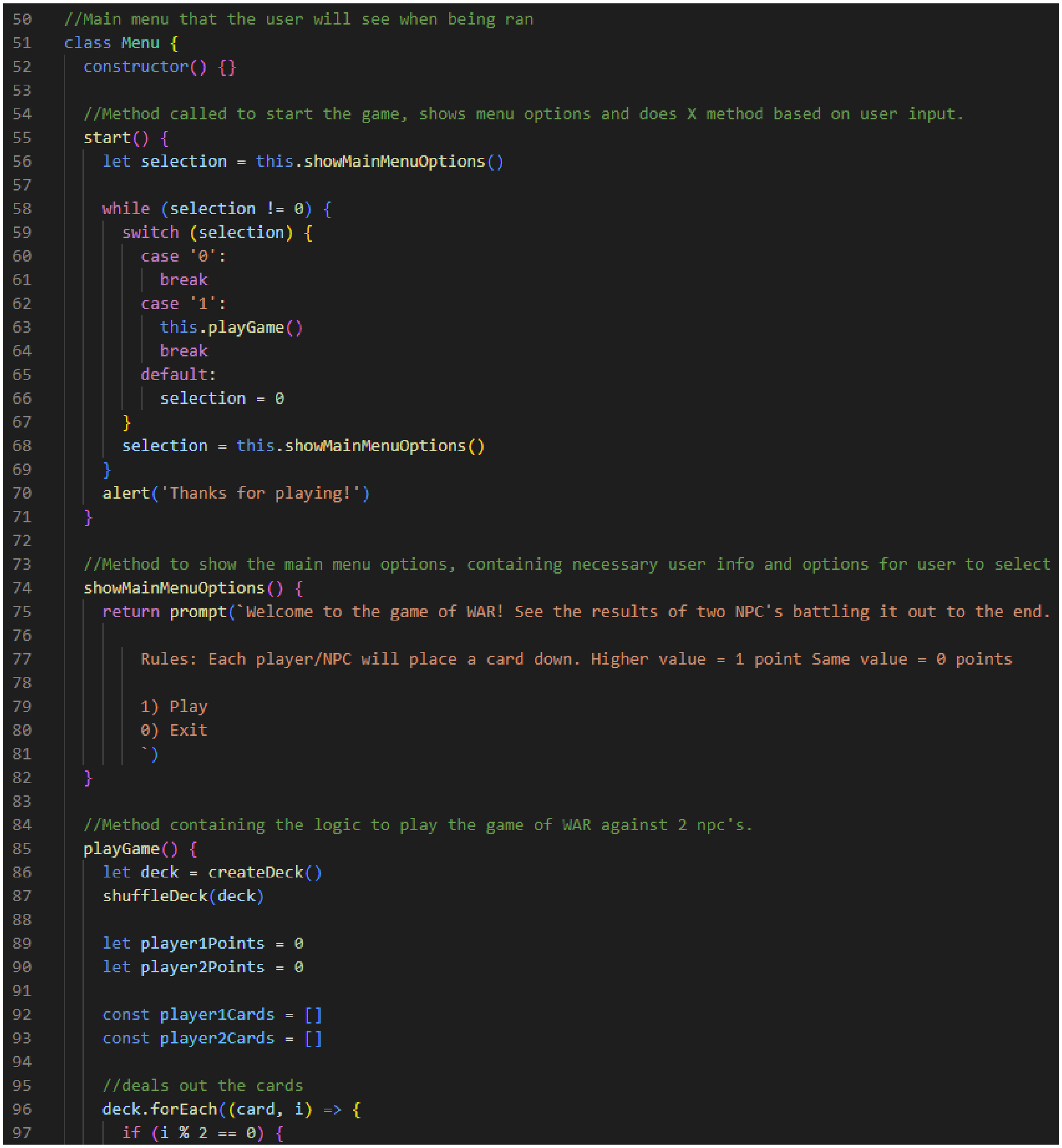
There are many versions of the game *WAR*, but in this version there are only 2 players and you don't need to do anything special when there is a tie on a round.

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:

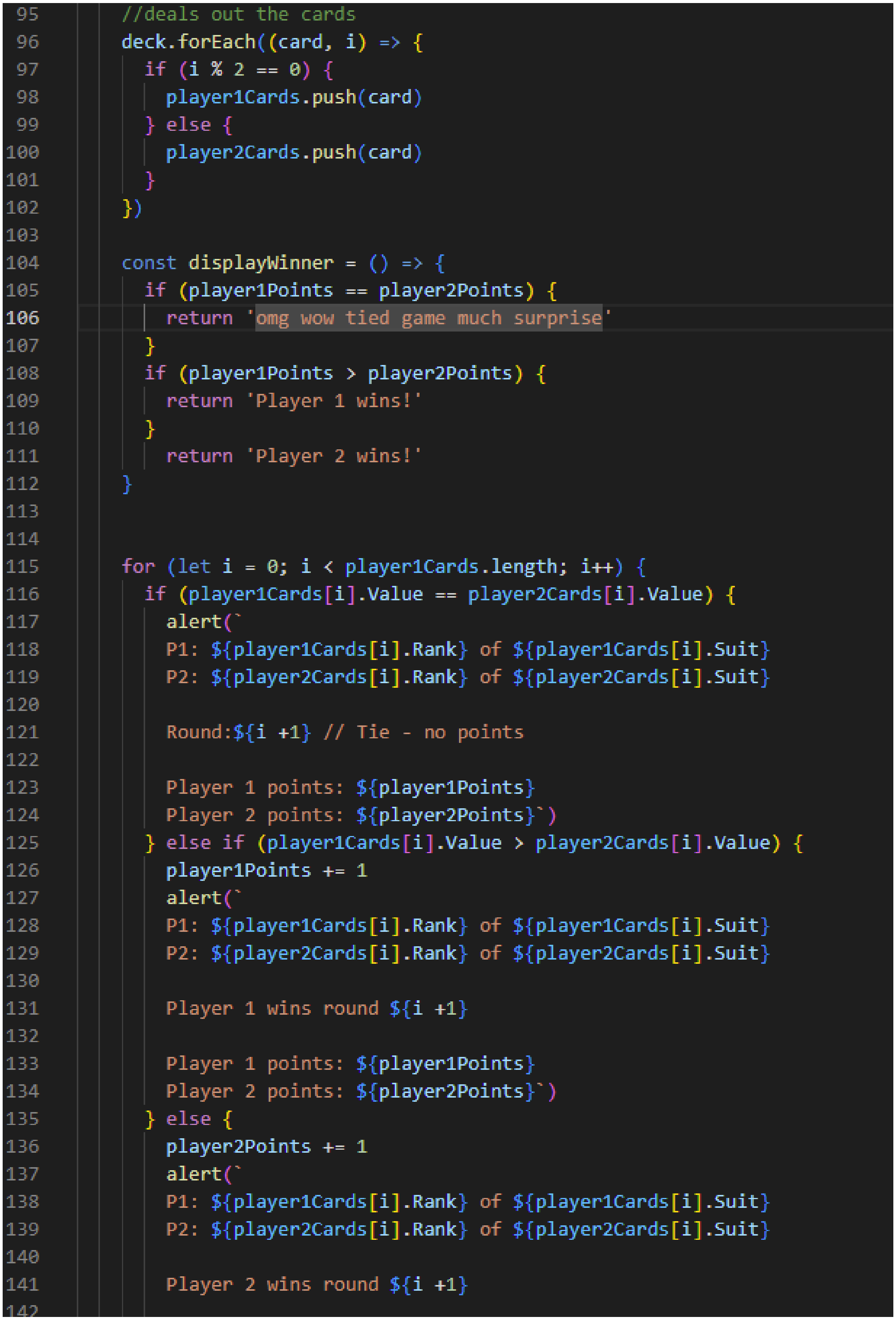
- Deal 26 Cards to two Players from a Deck.
- Iterate through the turns where each Player plays a Card



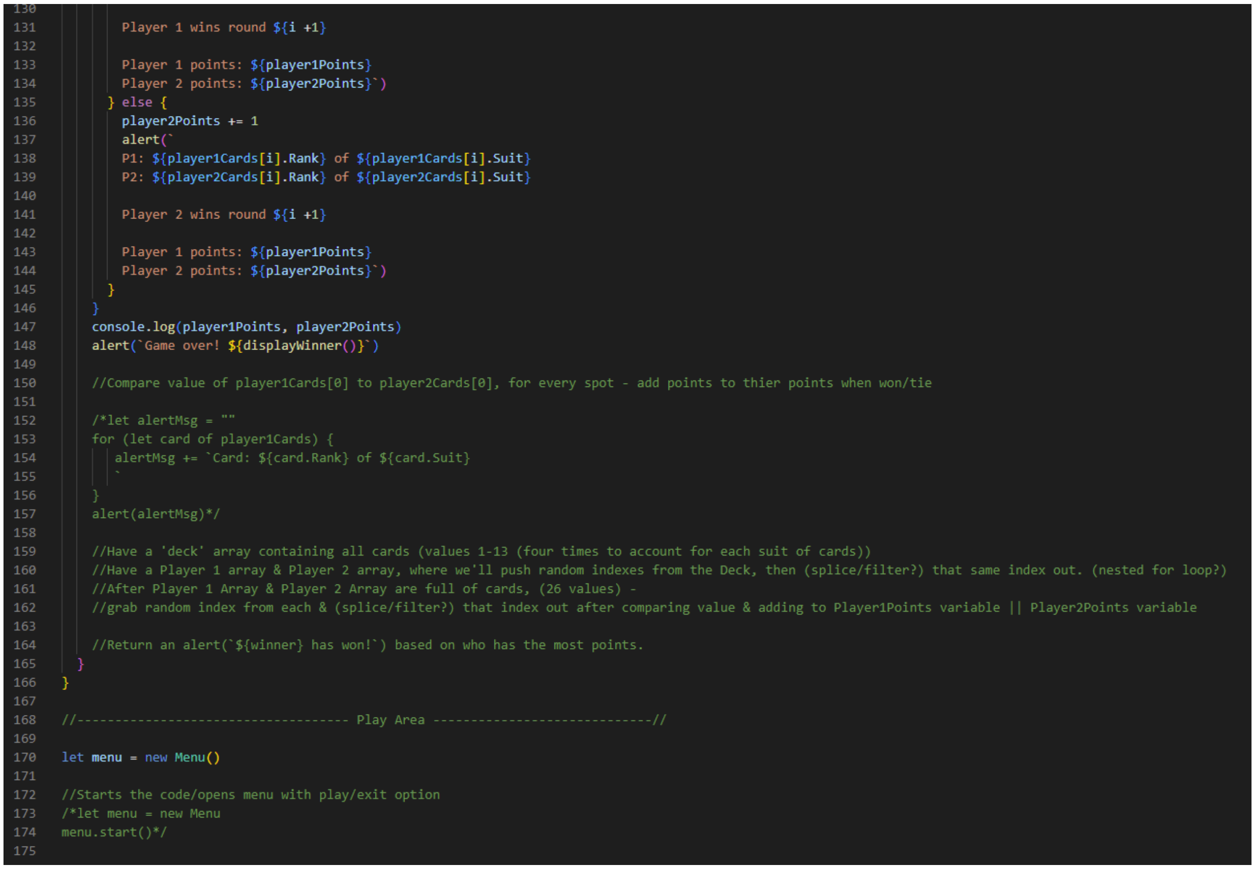
```javascript
//Game of WAR

/*- 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
o Ties result in zero points for either Player
- After all cards have been played, display the score.
Write a Unit Test using Mocha and Chai for at least one of the functions you write.
*/

//factory function that creates an ordered deck
const createDeck = () => {
  const suits = ['(♥ Hearts)', '(♠ Spades)', '(♦ Diamonds)', '(♣ Clubs)']
  const ranks = [
    'Ace',
    '2',
    '3',
    '4',
    '5',
    '6',
    '7',
    '8',
    '9',
    '10',
    'Jack',
    'Queen',
    'King',
  ]

  const deck = []
  suits.forEach((Suit) => {
    ranks.forEach((Rank, i) => {
      deck.push({ Suit, Rank, Value: i + 1 })
    })
  })

  return deck
}

//shuffles a deck
const shuffleDeck = (arr) => {
  for (let i = arr.length - 1; i > 0; i--) {
    let j = Math.floor(Math.random() * i)
    let temp = arr[i]
    arr[i] = arr[j]
    arr[j] = temp
  }
}

```



```
//Main menu that the user will see when being ran
class Menu {
  constructor() {}

  //Method called to start the game, shows menu options and does X method based on user input.
  start() {
    let selection = this.showMainMenuOptions()

    while (selection != 0) {
      switch (selection) {
        case '0':
          break
        case '1':
          this.playGame()
          break
        default:
          selection = 0
      }
      selection = this.showMainMenuOptions()
    }
    alert('Thanks for playing!')
  }

  //Method to show the main menu options, containing necessary user info and options for user to select
  showMainMenuOptions() {
    return prompt(`Welcome to the game of WAR! See the results of two NPC's battling it out to the end.

        Rules: Each player/NPC will place a card down. Higher value = 1 point Same value = 0 points

        1) Play
        0) Exit
        `)
  }

  //Method containing the logic to play the game of WAR against 2 npc's.
  playGame() {
    let deck = createDeck()
    shuffleDeck(deck)

    let player1Points = 0
    let player2Points = 0

    const player1Cards = []
    const player2Cards = []

    //deals out the cards
    deck.forEach((card, i) => {
      if (i % 2 == 0) {
```

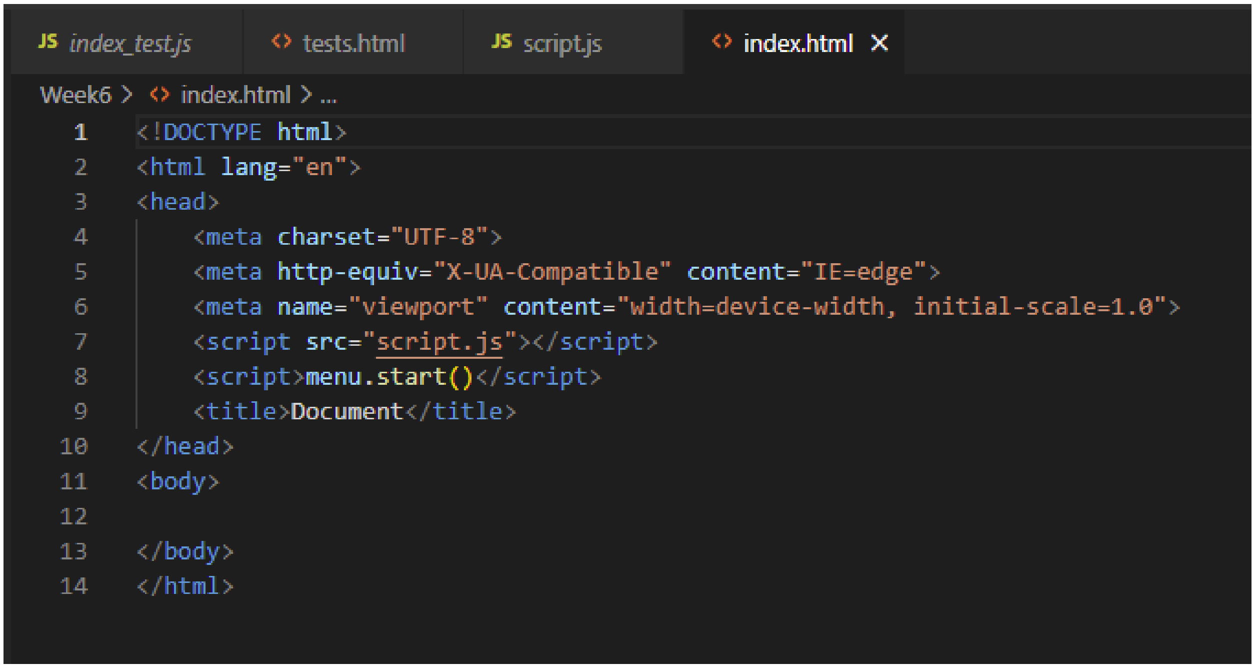


```
    //deals out the cards
    deck.forEach((card, i) => {
      if (i % 2 == 0) {
        player1Cards.push(card)
      } else {
        player2Cards.push(card)
      }
    })

    const displayWinner = () => {
      if (player1Points == player2Points) {
        return 'omg wow tied game much surprise'
      }
      if (player1Points > player2Points) {
        return 'Player 1 wins!'
      }
        return 'Player 2 wins!'
    }


    for (let i = 0; i < player1Cards.length; i++) {
      if (player1Cards[i].Value == player2Cards[i].Value) {
        alert(`
        P1: ${player1Cards[i].Rank} of ${player1Cards[i].Suit}
        P2: ${player2Cards[i].Rank} of ${player2Cards[i].Suit}

        Round:${i +1} // Tie - no points

        Player 1 points: ${player1Points}
        Player 2 points: ${player2Points}`)
      } else if (player1Cards[i].Value > player2Cards[i].Value) {
        player1Points += 1
        alert(`
        P1: ${player1Cards[i].Rank} of ${player1Cards[i].Suit}
        P2: ${player2Cards[i].Rank} of ${player2Cards[i].Suit}

        Player 1 wins round ${i +1}

        Player 1 points: ${player1Points}
        Player 2 points: ${player2Points}`)
      } else {
        player2Points += 1
        alert(`
        P1: ${player1Cards[i].Rank} of ${player1Cards[i].Suit}
        P2: ${player2Cards[i].Rank} of ${player2Cards[i].Suit}

        Player 2 wins round ${i +1}

```



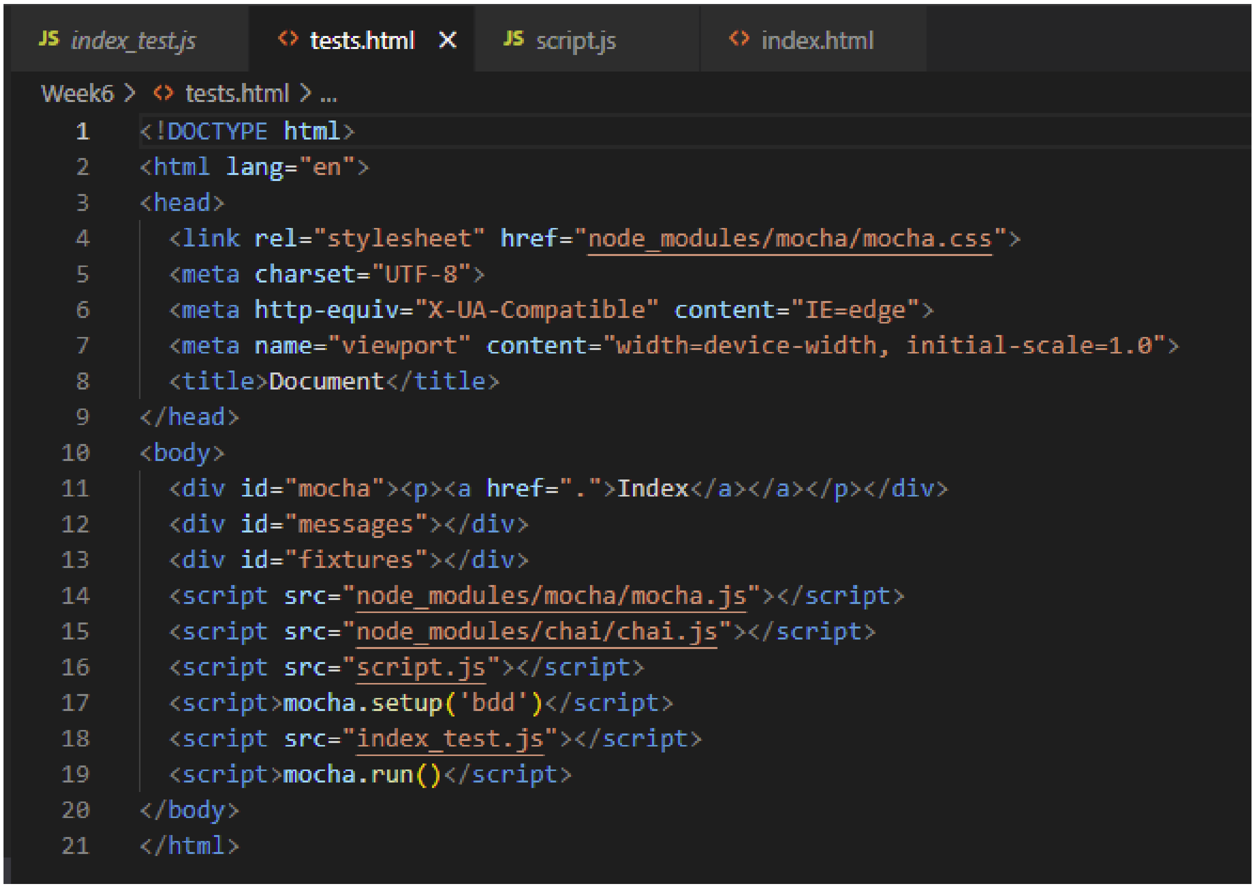
```
        Player 1 wins round ${i +1}

        Player 1 points: ${player1Points}
        Player 2 points: ${player2Points}`)
      } else {
        player2Points += 1
        alert(`
        P1: ${player1Cards[i].Rank} of ${player1Cards[i].Suit}
        P2: ${player2Cards[i].Rank} of ${player2Cards[i].Suit}

        Player 2 wins round ${i +1}

        Player 1 points: ${player1Points}
        Player 2 points: ${player2Points}`)
      }
    }
    console.log(player1Points, player2Points)
    alert(`Game over! ${displayWinner()}`)

    //Compare value of player1Cards[0] to player2Cards[0], for every spot - add points to thier points when won/tie

    /*let alertMsg = ""
    for (let card of player1Cards) {
       alertMsg += `Card: ${card.Rank} of ${card.Suit}
       `
    }
    alert(alertMsg)*/

    //Have a 'deck' array containing all cards (values 1-13 (four times to account for each suit of cards))
    //Have a Player 1 array & Player 2 array, where we'll push random indexes from the Deck, then (splice/filter?) that same index out. (nested for loop?)
    //After Player 1 Array & Player 2 Array are full of cards, (26 values) -
    //grab random index from each & (splice/filter?) that index out after comparing value & adding to Player1Points variable || Player2Points variable

    //Return an alert(`${winner} has won!`) based on who has the most points.
  }
}

//---------------------------------- Play Area ----------------------------//

let menu = new Menu()

//Starts the code/opens menu with play/exit option
/*let menu = new Menu
menu.start()*/
```

PAGE FOR WEB APP



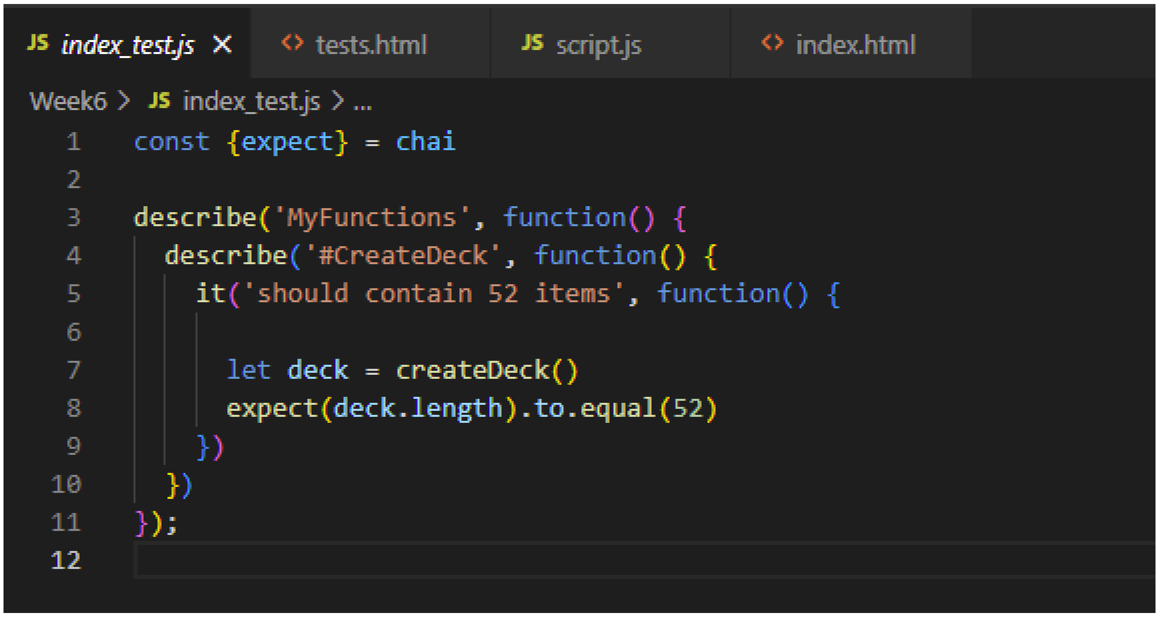
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <script src="script.js"></script>
    <script>menu.start()</script>
    <title>Document</title>
</head>
<body>

</body>
</html>
```

HTML PAGE FOR TESTS



```html
<!DOCTYPE html>
<html lang="en">
<head>
  <link rel="stylesheet" href="node_modules/mocha/mocha.css">
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  <div id="mocha"><p><a href=".">Index</a></a></p></div>
  <div id="messages"></div>
  <div id="fixtures"></div>
  <script src="node_modules/mocha/mocha.js"></script>
  <script src="node_modules/chai/chai.js"></script>
  <script src="script.js"></script>
  <script>mocha.setup('bdd')</script>
  <script src="index_test.js"></script>
  <script>mocha.run()</script>
</body>
</html>
```

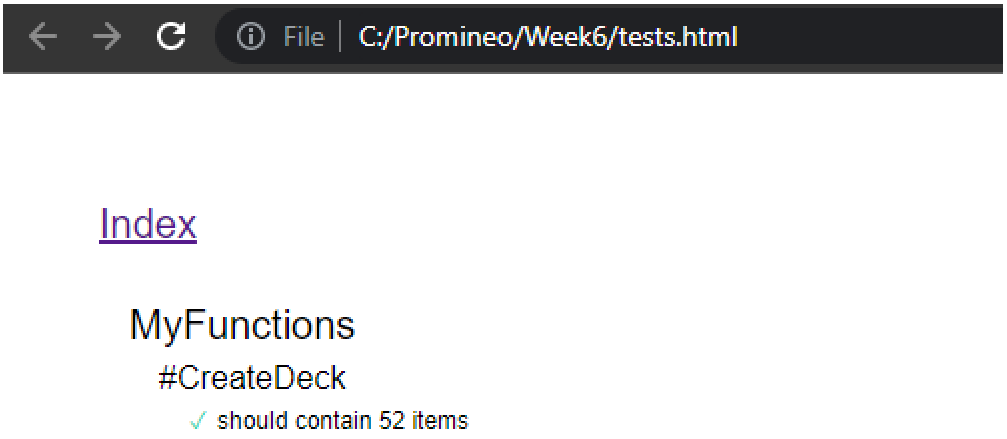
SCRIPT FOR TESTS



```javascript
const {expect} = chai

describe('MyFunctions', function() {
  describe('#CreateDeck', function() {
    it('should contain 52 items', function() {

      let deck = createDeck()
      expect(deck.length).to.equal(52)
    })
  })
});

```

Screenshots of Running Application:

TEST



C:/Promineo/Week6/tests.html

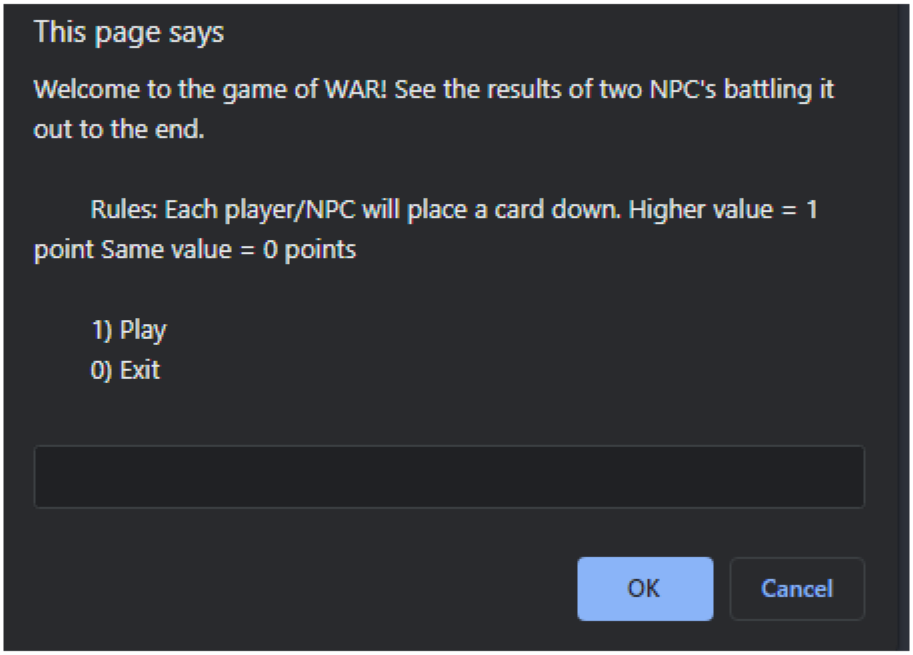
Index

MyFunctions

#CreateDeck

✓ should contain 52 items

CODE

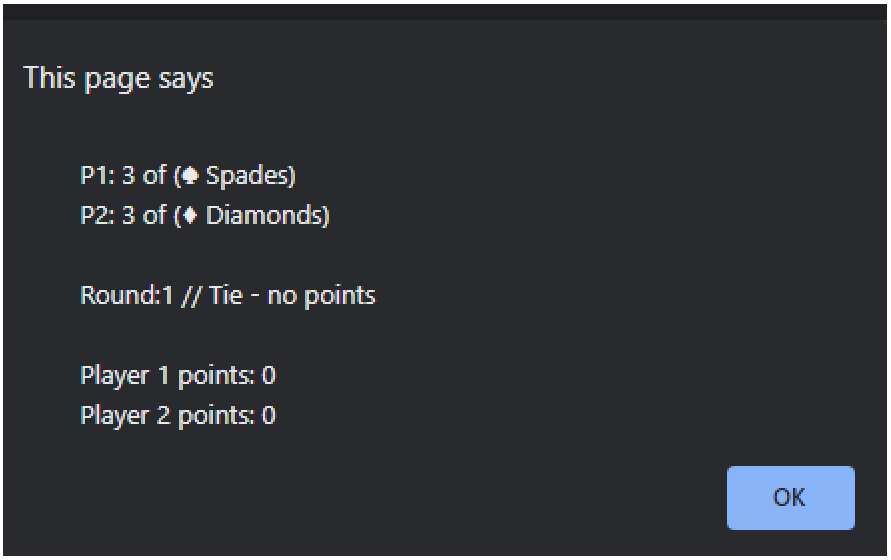


This page says

Welcome to the game of WAR! See the results of two NPC's battling it out to the end.

Rules: Each player/NPC will place a card down. Higher value = 1 point Same value = 0 points

1) Play
0) Exit

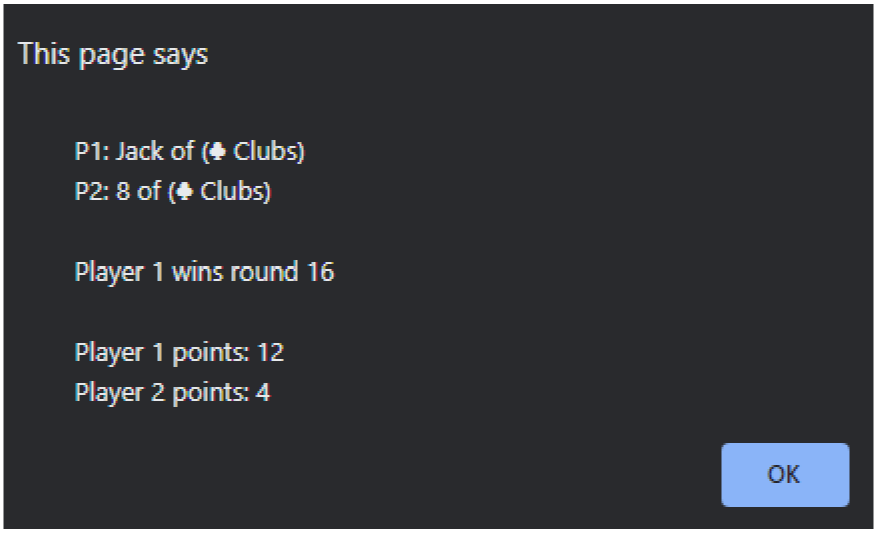


This page says

P1: 3 of (♠ Spades)
P2: 3 of (♦ Diamonds)

Round:1 // Tie - no points

Player 1 points: 0
Player 2 points: 0

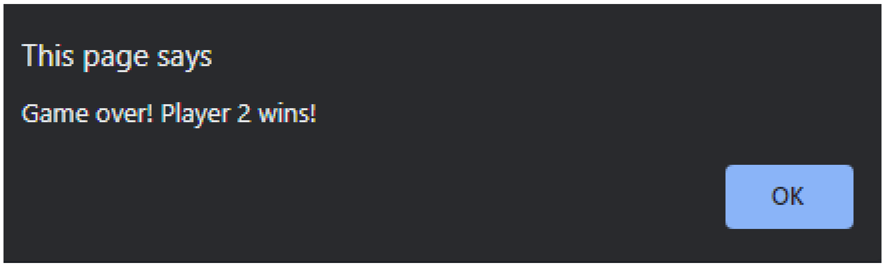


This page says

P1: Jack of (♣ Clubs)
P2: 8 of (♣ Clubs)

Player 1 wins round 16

Player 1 points: 12
Player 2 points: 4



This page says

Game over! Player 2 wins!

Page goes back to the start after the games over.

URL to GitHub Repository:

https://github.com/ConwayCJ/Week6