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*.

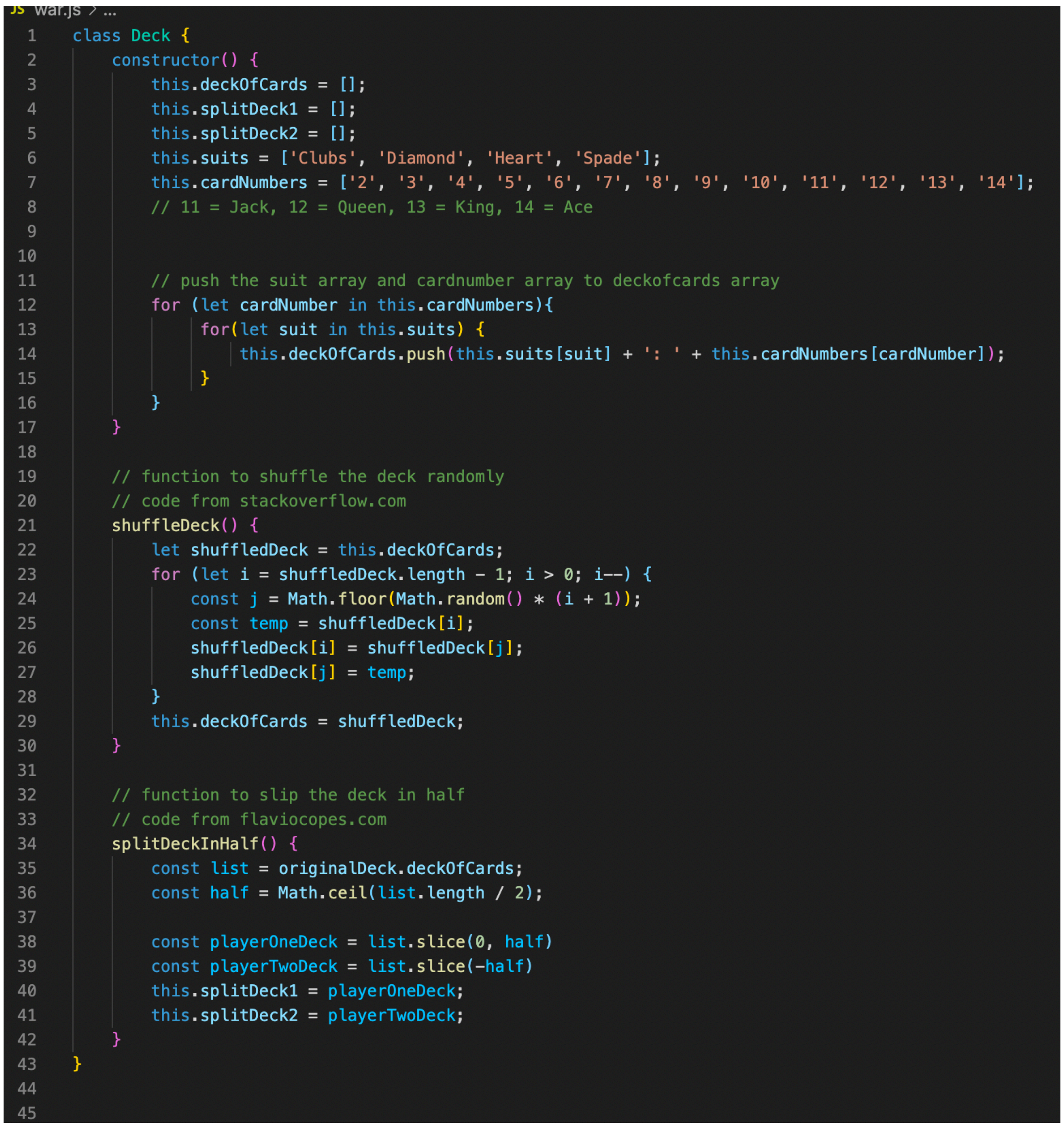
Think about how you would build this project and write your plan down. Consider classes such as Card, Deck, and Player and what fields and methods they might each have. You can implement the game however you'd like (i.e. printing to the console, using alert, or some other way). The completed project should, when ran, 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 either Player
- After all cards have been played, display the score.

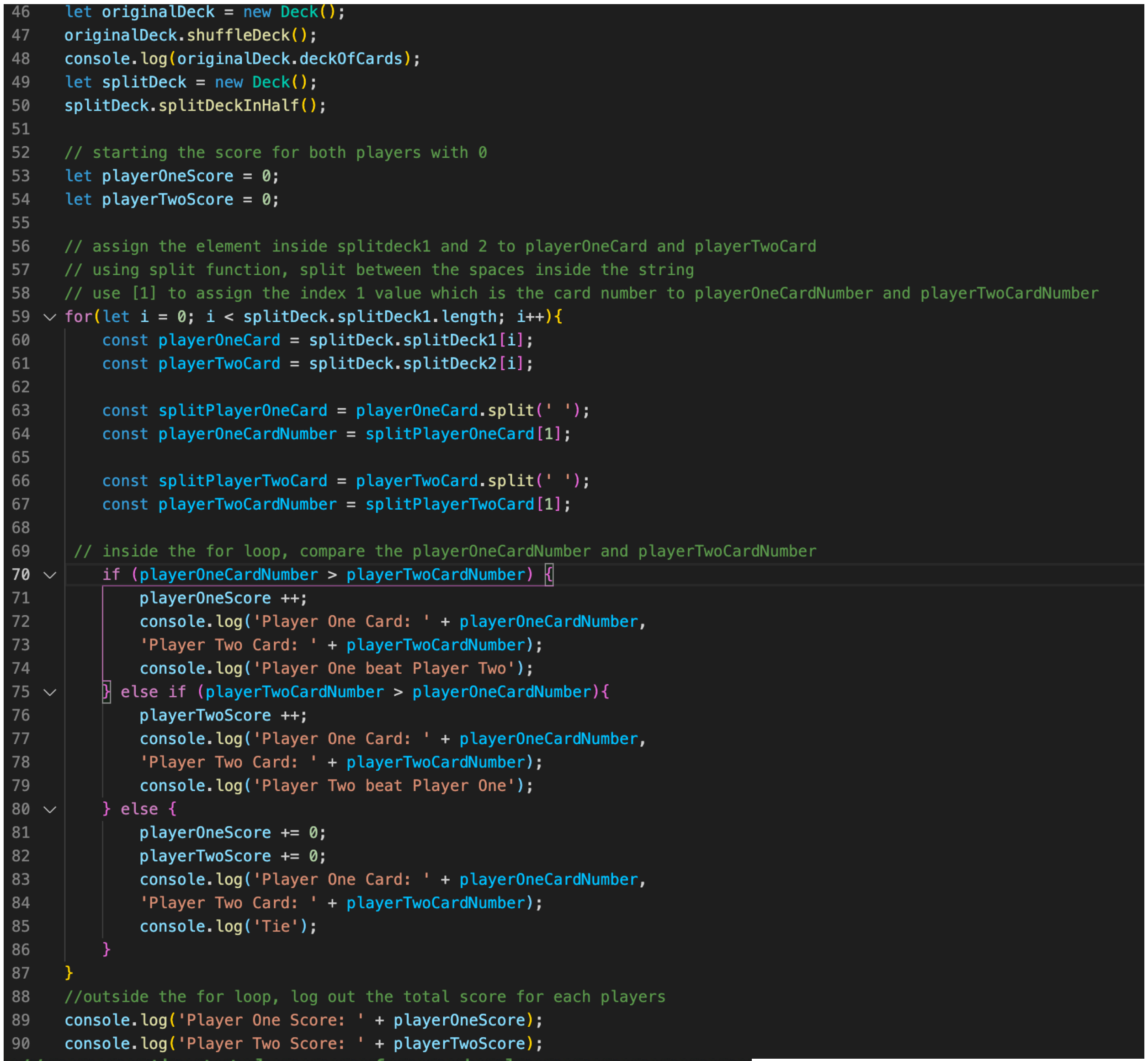
Write a Unit Test using Mocha and Chai for at least one of the functions you write.

Screenshots of Code:

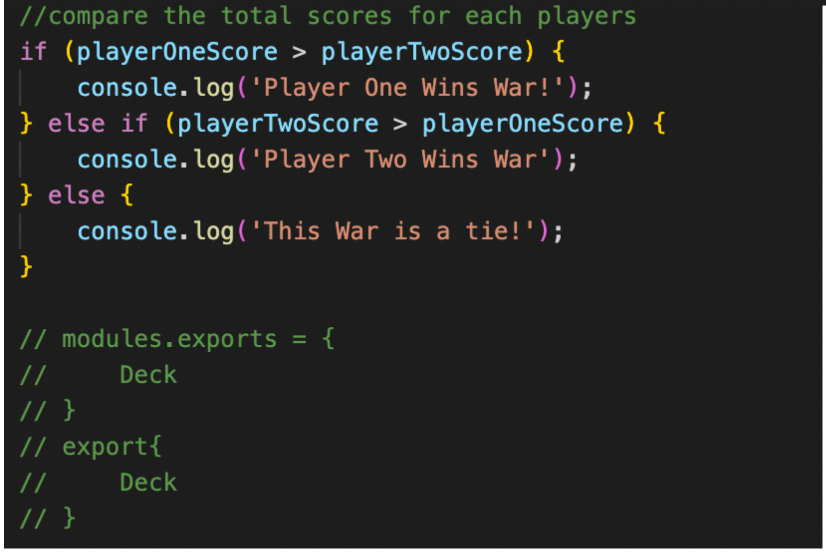


JS war.js > ...

```javascript
class Deck {
    constructor() {
        this.deckOfCards = [];
        this.splitDeck1 = [];
        this.splitDeck2 = [];
        this.suits = ['Clubs', 'Diamond', 'Heart', 'Spade'];
        this.cardNumbers = ['2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14'];
        // 11 = Jack, 12 = Queen, 13 = King, 14 = Ace


        // push the suit array and cardnumber array to deckofcards array
        for (let cardNumber in this.cardNumbers){
            for(let suit in this.suits) {
                this.deckOfCards.push(this.suits[suit] + ': ' + this.cardNumbers[cardNumber]);
            }
        }
    }

    // function to shuffle the deck randomly
    // code from stackoverflow.com
    shuffleDeck() {
        let shuffledDeck = this.deckOfCards;
        for (let i = shuffledDeck.length - 1; i > 0; i--) {
            const j = Math.floor(Math.random() * (i + 1));
            const temp = shuffledDeck[i];
            shuffledDeck[i] = shuffledDeck[j];
            shuffledDeck[j] = temp;
        }
        this.deckOfCards = shuffledDeck;
    }

    // function to slip the deck in half
    // code from flaviocopes.com
    splitDeckInHalf() {
        const list = originalDeck.deckOfCards;
        const half = Math.ceil(list.length / 2);

        const playerOneDeck = list.slice(0, half)
        const playerTwoDeck = list.slice(-half)
        this.splitDeck1 = playerOneDeck;
        this.splitDeck2 = playerTwoDeck;
    }
}
```

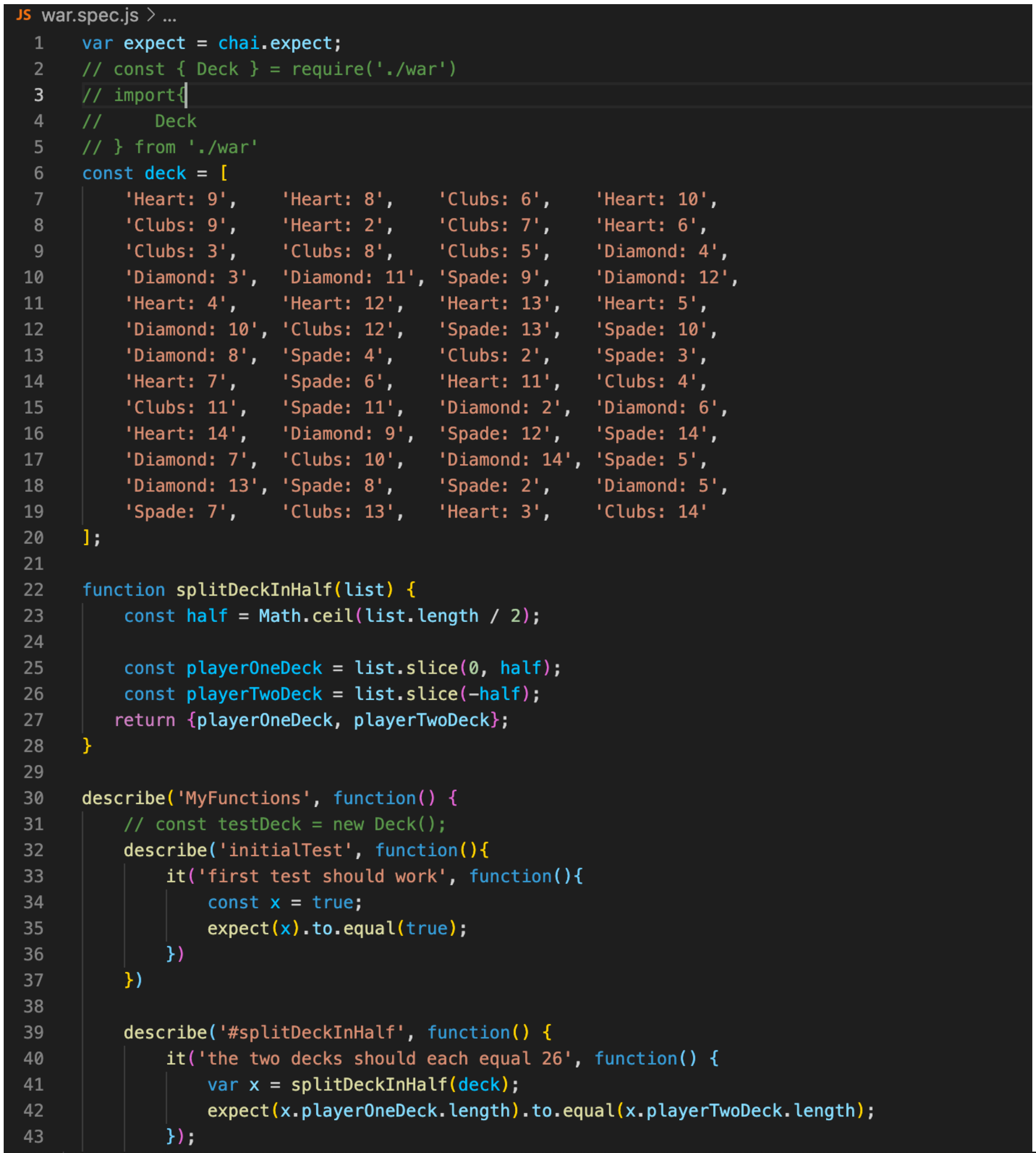


```javascript
let originalDeck = new Deck();
originalDeck.shuffleDeck();
console.log(originalDeck.deckOfCards);
let splitDeck = new Deck();
splitDeck.splitDeckInHalf();

// starting the score for both players with 0
let playerOneScore = 0;
let playerTwoScore = 0;

// assign the element inside splitdeck1 and 2 to playerOneCard and playerTwoCard
// using split function, split between the spaces inside the string
// use [1] to assign the index 1 value which is the card number to playerOneCardNumber and playerTwoCardNumber
for(let i = 0; i < splitDeck.splitDeck1.length; i++){
    const playerOneCard = splitDeck.splitDeck1[i];
    const playerTwoCard = splitDeck.splitDeck2[i];

    const splitPlayerOneCard = playerOneCard.split(' ');
    const playerOneCardNumber = splitPlayerOneCard[1];

    const splitPlayerTwoCard = playerTwoCard.split(' ');
    const playerTwoCardNumber = splitPlayerTwoCard[1];

 // inside the for loop, compare the playerOneCardNumber and playerTwoCardNumber
    if (playerOneCardNumber > playerTwoCardNumber) {
        playerOneScore ++;
        console.log('Player One Card: ' + playerOneCardNumber,
        'Player Two Card: ' + playerTwoCardNumber);
        console.log('Player One beat Player Two');
    } else if (playerTwoCardNumber > playerOneCardNumber){
        playerTwoScore ++;
        console.log('Player One Card: ' + playerOneCardNumber,
        'Player Two Card: ' + playerTwoCardNumber);
        console.log('Player Two beat Player One');
    } else {
        playerOneScore += 0;
        playerTwoScore += 0;
        console.log('Player One Card: ' + playerOneCardNumber,
        'Player Two Card: ' + playerTwoCardNumber);
        console.log('Tie');
    }
}
//outside the for loop, log out the total score for each players
console.log('Player One Score: ' + playerOneScore);
console.log('Player Two Score: ' + playerTwoScore);
```



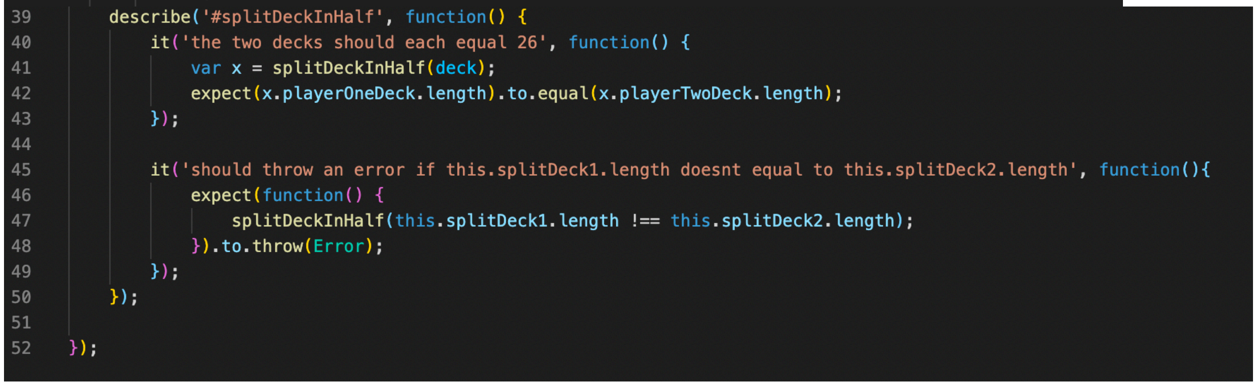
```javascript
//compare the total scores for each players
if (playerOneScore > playerTwoScore) {
    console.log('Player One Wins War!');
} else if (playerTwoScore > playerOneScore) {
    console.log('Player Two Wins War');
} else {
    console.log('This War is a tie!');
}

// modules.exports = {
//      Deck
// }
// export{
//      Deck
// }
```

PROMINEO TECH



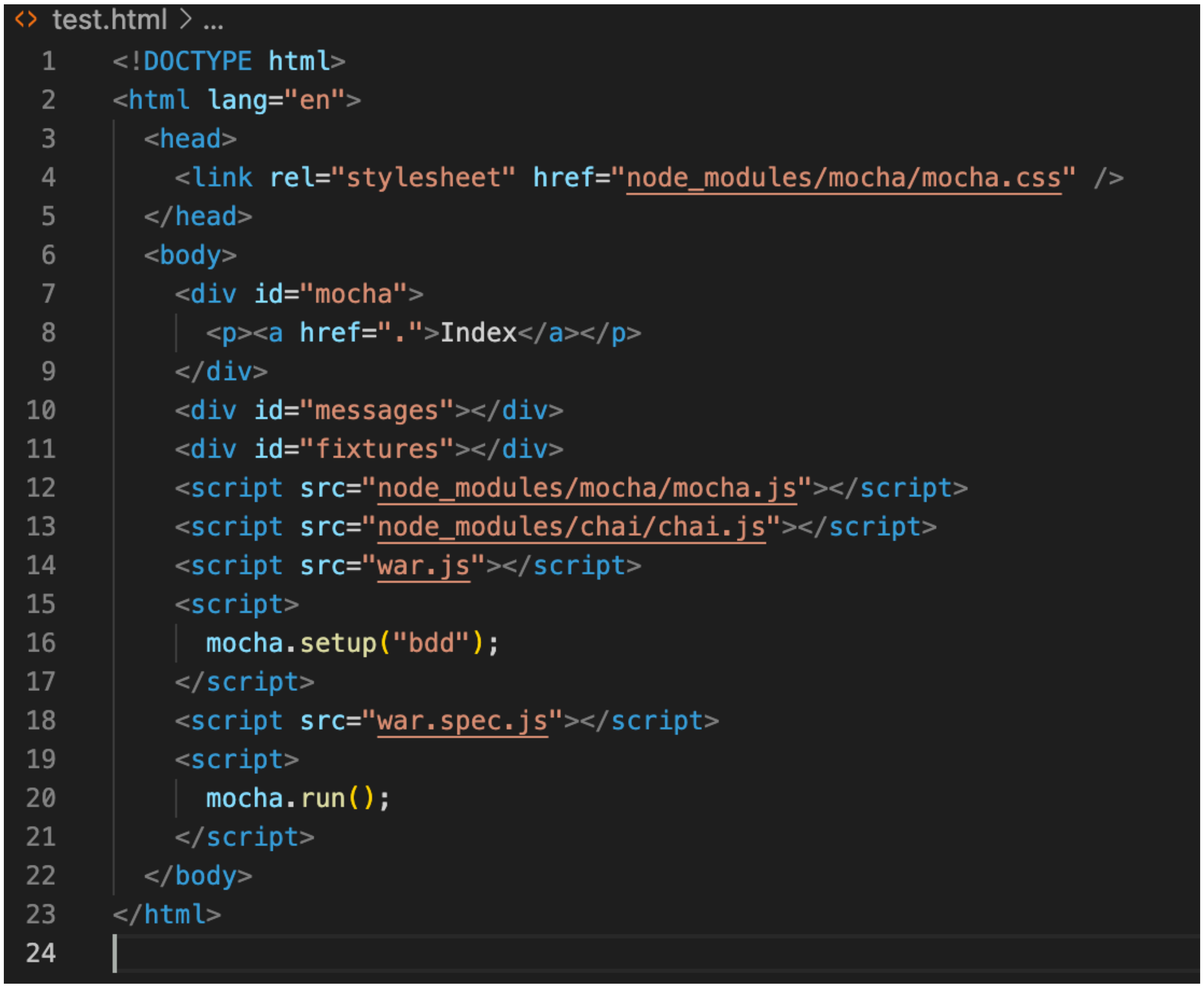
JS war.spec.js > ...

```javascript
var expect = chai.expect;
// const { Deck } = require('./war')
// import{
//     Deck
// } from './war'
const deck = [
    'Heart: 9',    'Heart: 8',    'Clubs: 6',    'Heart: 10',
    'Clubs: 9',    'Heart: 2',    'Clubs: 7',    'Heart: 6',
    'Clubs: 3',    'Clubs: 8',    'Clubs: 5',    'Diamond: 4',
    'Diamond: 3',  'Diamond: 11', 'Spade: 9',    'Diamond: 12',
    'Heart: 4',    'Heart: 12',   'Heart: 13',   'Heart: 5',
    'Diamond: 10', 'Clubs: 12',   'Spade: 13',   'Spade: 10',
    'Diamond: 8',  'Spade: 4',    'Clubs: 2',    'Spade: 3',
    'Heart: 7',    'Spade: 6',    'Heart: 11',   'Clubs: 4',
    'Clubs: 11',   'Spade: 11',   'Diamond: 2',  'Diamond: 6',
    'Heart: 14',   'Diamond: 9',  'Spade: 12',   'Spade: 14',
    'Diamond: 7',  'Clubs: 10',   'Diamond: 14', 'Spade: 5',
    'Diamond: 13', 'Spade: 8',    'Spade: 2',    'Diamond: 5',
    'Spade: 7',    'Clubs: 13',   'Heart: 3',    'Clubs: 14'
];

function splitDeckInHalf(list) {
    const half = Math.ceil(list.length / 2);

    const playerOneDeck = list.slice(0, half);
    const playerTwoDeck = list.slice(-half);
   return {playerOneDeck, playerTwoDeck};
}

describe('MyFunctions', function() {
    // const testDeck = new Deck();
    describe('initialTest', function(){
        it('first test should work', function(){
            const x = true;
            expect(x).to.equal(true);
        })
    })

    describe('#splitDeckInHalf', function() {
        it('the two decks should each equal 26', function() {
            var x = splitDeckInHalf(deck);
            expect(x.playerOneDeck.length).to.equal(x.playerTwoDeck.length);
        });
```



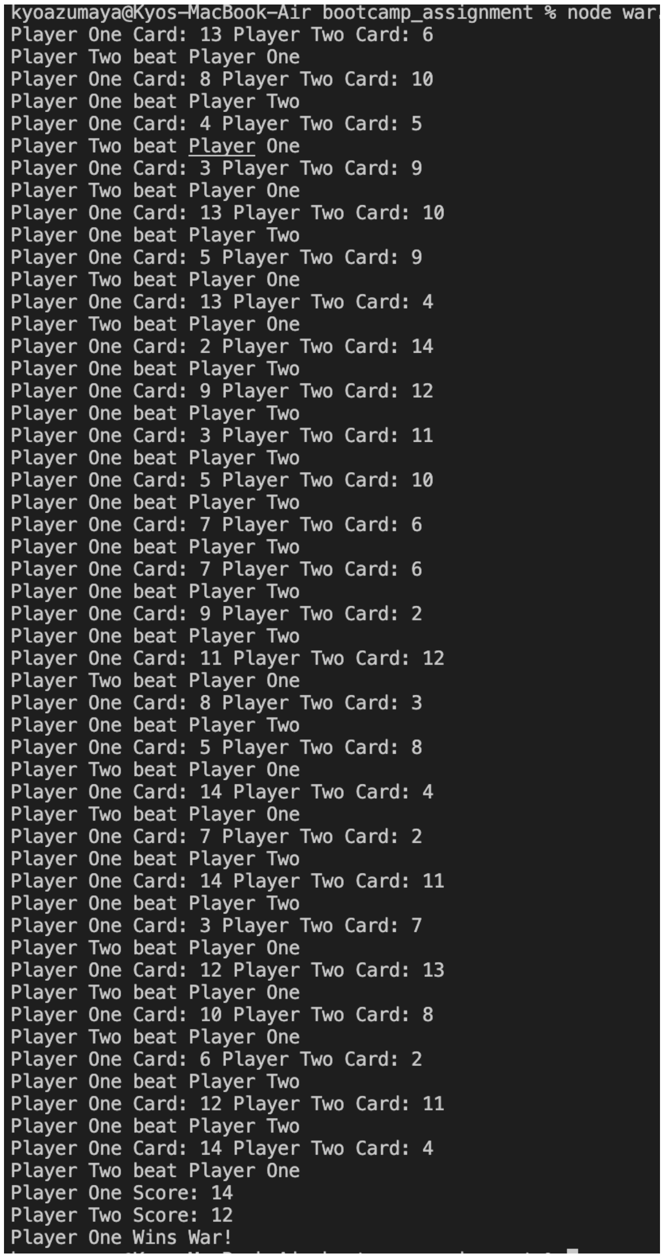
```javascript
    describe('#splitDeckInHalf', function() {
        it('the two decks should each equal 26', function() {
            var x = splitDeckInHalf(deck);
            expect(x.playerOneDeck.length).to.equal(x.playerTwoDeck.length);
        });
```

```javascript

        it('should throw an error if this.splitDeck1.length doesnt equal to this.splitDeck2.length', function(){
            expect(function() {
                splitDeckInHalf(this.splitDeck1.length !== this.splitDeck2.length);
            }).to.throw(Error);
        });
    });

});
```

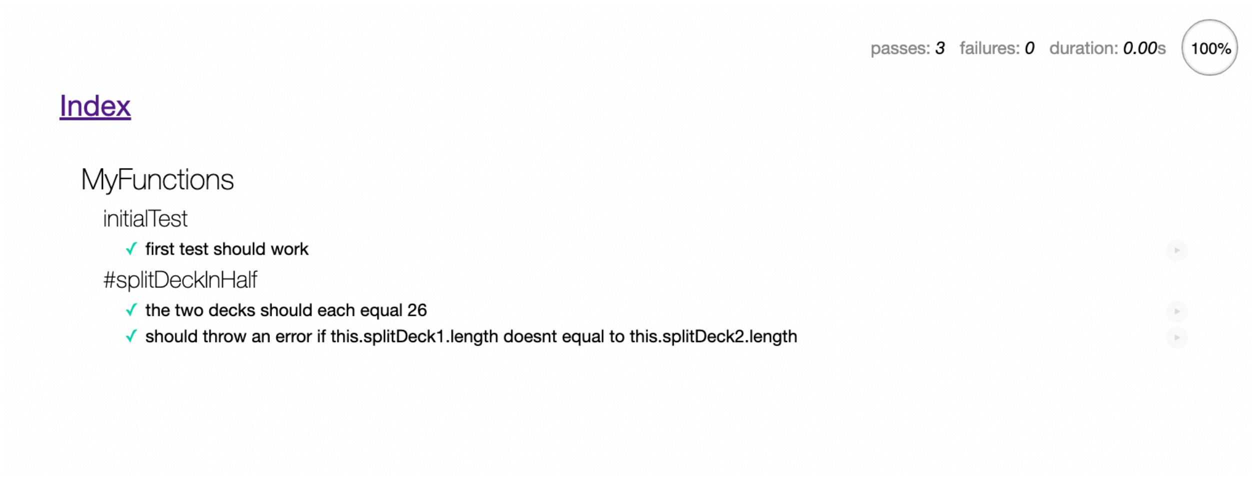

test.html > ...

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <link rel="stylesheet" href="node_modules/mocha/mocha.css" />
  </head>
  <body>
    <div id="mocha">
      <p><a href=".">Index</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="war.js"></script>
    <script>
      mocha.setup("bdd");
    </script>
    <script src="war.spec.js"></script>
    <script>
      mocha.run();
    </script>
  </body>
</html>

```

Screenshots of Running Application:



```
kyoazumaya@Kyos-MacBook-Air bootcamp_assignment % node war
Player One Card: 13 Player Two Card: 6
Player Two beat Player One
Player One Card: 8 Player Two Card: 10
Player One beat Player Two
Player One Card: 4 Player Two Card: 5
Player Two beat Player One
Player One Card: 3 Player Two Card: 9
Player Two beat Player One
Player One Card: 13 Player Two Card: 10
Player One beat Player Two
Player One Card: 5 Player Two Card: 9
Player Two beat Player One
Player One Card: 13 Player Two Card: 4
Player Two beat Player One
Player One Card: 2 Player Two Card: 14
Player One beat Player Two
Player One Card: 9 Player Two Card: 12
Player One beat Player Two
Player One Card: 3 Player Two Card: 11
Player One beat Player Two
Player One Card: 5 Player Two Card: 10
Player One beat Player Two
Player One Card: 7 Player Two Card: 6
Player One beat Player Two
Player One Card: 7 Player Two Card: 6
Player One beat Player Two
Player One Card: 9 Player Two Card: 2
Player One beat Player Two
Player One Card: 11 Player Two Card: 12
Player Two beat Player One
Player One Card: 8 Player Two Card: 3
Player One beat Player Two
Player One Card: 5 Player Two Card: 8
Player Two beat Player One
Player One Card: 14 Player Two Card: 4
Player Two beat Player One
Player One Card: 7 Player Two Card: 2
Player One beat Player Two
Player One Card: 14 Player Two Card: 11
Player One beat Player Two
Player One Card: 3 Player Two Card: 7
Player Two beat Player One
Player One Card: 12 Player Two Card: 13
Player Two beat Player One
Player One Card: 10 Player Two Card: 8
Player Two beat Player One
Player One Card: 6 Player Two Card: 2
Player One beat Player Two
Player One Card: 12 Player Two Card: 11
Player One beat Player Two
Player One Card: 14 Player Two Card: 4
Player Two beat Player One
Player One Score: 14
Player Two Score: 12
Player One Wins War!
```



```
passes: 3   failures: 0   duration: 0.00s   100%

Index

MyFunctions
  initialTest
    ✓ first test should work
  #splitDeckInHalf
    ✓ the two decks should each equal 26
    ✓ should throw an error if this.splitDeck1.length doesnt equal to this.splitDeck2.length
```

URL to GitHub Repository:

https://github.com/KyoAzumaya/warCardGame.git