```javascript
1  // we create a class for what the card would hold which is the suit, the
   ranking, and value within the contructor meaning where we construct the parts
   of the cards
2
3  import card ./
4
5  export class Card [] {
6    constructor(rank,suit,value)
7      this.suit = suit;
8      this.rank = rank;
9      this.value = value;
10   }
11
12  // we create another class for the deck it will at "this.cards" insert the
   set of cards
13
14   class Deck {
15   constructor(){
16     this.cards = [];
17   };
18
19   // at getCards the player can choose a card from the deck which will be
   returned to them.
20
21   getCards() {
22     return this.cards;
23   }
24
25   //at buildDeck we want it to run and populate through the deck and also
   shuffle
26   //we return the cards right after
27
28   buildDeck() {
29     this.populate();
30     this.shuffle();
31     return this.cards;
32   }
33  //Populate is defined as the cards names, ranking and value we have them all
   listed here.
34  //We define the suits as the type of cards, we define the ranks as the tier,
   and we define the value as the number it holds
35
36   populate() {
37     const suits = ['Spades', 'Hearts', 'Clubs','Diamonds']
38     const ranks = ['A','2','3','4','5','6','7','8','9','10','J','Q','K'];
39     const values = ['1','2','3','4','5','6','7','8','9','10','11','12','13'];
40
41     // let the indx be equal to 0, then i if I is less then the length og the
   suit array ('Spades',etcs) it will increment by 1
42     //Same goes with the ranks
43     // we want to loop through the amount of type of cards and also loop
   through the ranks of the cards then push the variable newcards as the rank
   suit and values.
44
45     for (let i = 0; i < suits.length;i++) {
46       for (let j = 0; j < ranks.length; j++) {
47         this.cards.push(newCard(ranks[j],suits[j],values[j]));
48       }
49     }
```

```javascript
50    }
51
52    // we want the deck to shuffle through the cards so we implement a
   statement thats says
53    // if the length of the cards is greater then 0 then the deck being
   shuffled with include sorting the cards,randomizing it by subtracting by 0.5
54
55    shuffle() {
56        if (this.cards.length > 0) {
57        const shuffleDeck = this.cards.sort(() => Math.random() - 0.5)
58        this.cards = [shuffleDeck];
59      }
60    }
61 }
62
63 //when it comes to the players we want to construct by using a constucter
   that would name the score,deck, and name of the player
64 //we write each variable to then display. By writing this.(name of
   variable,action,object) we can then equal it to the name,object or action.
65
66 class Player {
67
68     constructor(name) {
69      this.playerName = name;
70      this.playerScore = 0;
71      this.playerDeck = [];
72     }
73     get name() {
74         return this.playerName;
75     }
76     get playerScore () {
77         return this.playerScore;
78     }
79     get deck(newDeck) {
80       if(Array.isArray(newDeck)) {
81         return this.playerDeck = newDeck;
82       }
83     }
84     set score(newScore) {
85       if (!isNaN(newScore)) {
86         this.playerScore = newScore;
87       }
88     }
89
90 }
91
92
93 //Now this is where we name off the Game it self and construct what we will
   be implementing and displaying
94 // we create a constructor to place the players and deck in that we created
   before
95
96 class WarGame {
97
98     constructor () {
99       this.players = [];
100       this.deck = [];
101     }
102
```

```javascript
103   //Heres where we actually start. This is where we actually implement all fo
      the variables,objects,actions that we created by making arrays using methods
      and listing off objects and names.
104   // We start off by displaying the tile where it says console.log,then we
      want it to display a prompt we do this by wriitng it out in brackets
105   //we implement a switch to list all of the options ^ means its an exit click
      on that key to initiate it and so on.
106   //at the end our input should equal our prompts
107
108     start() {
109
110
111       console.log('*** War Game ***')
112       let input = prompt('0 - Exit; 1 - Play; 2 - Look at game instructions');
113        while (input != '^') {
114           switch (input) {
115               case '^':
116                  exit;S
117               case '1':
118                 this.createGame();
119               case '2':
120                 this.instructions();
121                 break;
122           }
123           input = prompt(`0 - Exit; 1 - Play; 2 - click here to learn how to
      play`):
124         }
125
126 }
127
128 //now that we've listed all the actions,named the players, and objects we
      need to initate the players.
129 //player 1 will equal to a new player same goes for two by using this.
      (variable name).
130 //we create a var called cards and equal it to a new that that will be built
      everytime
131 //we use the slice method to split my deck in half. Then we connsole log the
      title (dealing hands)
132 //we create a statement if we let equal 0 and  i is less than this.(our
      players) but also if ourplayers,deck and value are great then the others
      players deck and value are we
133 //we will get an output of our players score being equal to 1
134 //let the winning hand equal it
135
136 createGame() {
137       this.players[0] = new player ('Player 1')
138       this.players[1] = new player ('Player 2')
139
140       const cards = new this.Deck().buildDeck();
141
142       this.players[0].deck = [cards.slice(0,26)];
143       this.players[1].deck = [cards.slice(26,52)];
144
145       console.log('****** Dealing Hands *******')
146       for (let i = 0; i < this.players[0].deck.length; i++) {
147           if (this.players[0].deck[i].value > this.players[i].deck[i].value) {
148               this.players[0].score +=1;
149
150           let winningHand = '${this.players[0].deck[i].rank} of
      ${this.players[0].deck[i].suit}';
```

```javascript
151            console.log('Player 1 won with a ${winnningHand}');
152            }
153        }
154 }
155
156     console.log('****** Hands Finished *****')
157
158
159     if(this.players[0].score > this.players[1].score) {
160     console.log('$(this.players[0].name.toUppercase()} WON THE WAR with a
    score of $(this.players[0].score}');
161     } else if (this.players[0].score < this.players[1].score) {
162     console.log('$(this.players[1].name.toUpperCase()} WON THE WAR with a
    score of $(this.players[1].score}');
163     } else {
164     console.log('Player one and Player tied');
165   }
166  }
167
168 // a const war game where it starts
169
170 const game = new WarGame();
171 game.start();
172
173
```