War Project

War is a simple card game involving two players. It uses a standard playing card deck (52 cards). The value of each card in decreasing order is: A K Q J T 9 8 7 6 5 4 3 2.

The objective of the game is to win all cards.

The deck is shuffled and divided evenly among the players, giving each a down stack.
On each round, each player reveals the **top card** of their deck – this is a "battle" – and the player with the **higher card takes** both of the cards played and moves them to the bottom of their stack.
If the **two cards played** are of equal value, then there is a "war".

> Both players place the next three cards of their pile face down, and then another card face-up. The owner of the higher face-up card wins the war and adds all ten cards on the table to the bottom of their deck. If the face-up cards are again equal then the battle repeats with another set of face-down/up cards. This repeats until one player's face-up card is higher than their opponent's

The game ends when wither player run out of cards, or when the player gives up.

Write a Python code to allow a player to play the game against the computer.

- Use Lists to store your decks.
- Use integers to represent the values of your cards
  Ace =1, Two = 2, Three = 3, Four = 4, Five = 5.... T = 10, J = 11, Q = 12, K = 13

Here is an online description of the game
https://www.bicyclecards.com/how-to-play/war/

You can play the game here
https://cardgames.io/war/

Use a **menu** with the following options:

1. **Play**- plays one round of war.
2. **Display Scores**-Displays the number of cards in each player’s deck
3. **Display Cards**-Displays the player’s cards in ascending order.
4. **Win Percentage**- Displays the percentage of battles the player has won
5. **Quit**- Quit the game

Implement the following functions:

1. **initializeDeck**: Initialize a deck of 52 cards. Use four of each of the value 1-13.
2. **shuffleEeck**: Take the newly created deck and shuffle it. (One way to shuffle is to swap the value two randomly selected locations-Repeat until the deck is fully shuffled-
3. **splitDeck**: Pass in the full shuffled deck, remove half of it place it into the new deck.
4. **sortDeck**-Returned a sorted deck that has been passed in. (Do not sort the actual deck, sort a copy of it and return it.)
5. **printDeck**- Prints the content of the deck in ascending order (You should call sortD in it)
6. **distributeCards**- Places battle card at the bottom of the winner’s deck. (loser loses a card to the winner; winner places their played card at the bottom of their deck)

7. **Battle**- Plays a single round. Compare top cards of each deck, determines the winner, calls distributeCards, and returns who won. Call the war method -when both cards are of equal value- to determine the winner. War repeats as long as there is a tie. Determines the winner, calls distributeCards(), and returns who won. If a player does not have enough cards to play a war, they lose.
8. **displayStatistics** – calculate and display the percentage of player wins.
9. **displayScore** – displays the number of cards in both decks.

Use what we learnt only, it might be tempting to use OOP, but we didn't learn it.
Run your code for multiple output
Document your code properly.
Paste your code, and multiple sample outputs in a word document, upload as a pdf.
Record a **video demonstration** of your code being run for possible full credit. Upload your videos to YouTube and share the link with me in the same document that has the code and the outputs.

Projects without demonstration will earn 10% of the total grade even if it is correct.
Do NOT use anything that we did not learn in the classroom. Do not use dictionary, use the list technique I demonstrated in the seminar. Do not use object-oriented programming yet.

Programming assignments are supposed to be solved gradually. You should not solve them in one day.

Cheaters will not succeed. Cheating is looking for the solution. Cheaters don't even try to solve the problem; cheaters will hurt themselves first.

Any cheating attempt will be reported to the university.

Be creative, you can, I believe in you.
I know what we covered
Start early, take baby steps in your solution
Enjoy what you are doing

Program with love and care for the code and for what you are doing