Lab 3.03 - War (Card Game)

Create a program that lets a user play a **simplified** version of the card game **'War'**. In this version, the users will share a single deck of cards and cards will not be added back to the deck after they have been played.

Video Explanation of the Card Game War

https://youtu.be/yX-jOVer758

Your game should

- start with a given shuffled deck variable (shuffle function comes from python's random library, more details below)
- ask for player1 and player2's names.
- have a function `player_turn`, with the contract shown below:

```
# Name: player_turn
# Purpose: takes in a player name,
#          draws/removes a card from the deck,
#          prints "user drew card x",
#          and returns the value
# Arguments: player_name as string, deck as list
# Returns: integer
```

- Jacks will be represented as 11, Queens will be represented as 12, Kings will be represented as 13, and Aces will be represented as 14. The suit does not matter.
- Create a function `card_name` to be used by `player_turn()`, that takes in an integer representing a drawn card, and returns a string that names the card. 2 prints as `"2"`, 3 is `"3"`, etc., but 11 is `"J"`, 12 is `"Q"`, 13 is `"K"`, and 14 is `"A"`.
- Make sure to write the contract for card_name()!
- Include a `while` loop that keeps the game running until there are no cards in the deck.
- If there is a tie, there is "war". Take the next two cards. Whoever wins that comparison gets all four cards (including the previous tied cards).
- If there is another tie, continue taking the next two cards until there a winner.
- The winner takes all the "war" cards.
- Keep track of the score.
- The player who takes the greatest number of cards wins.
- Declare the name of the winner and final score at the end of the game.

Sample Output

```
Player 1's name: Pat Player 2's name: Sam

Pat drew 8 Sam drew 9 Sam has high card Pat: 0 Sam: 2
Pat drew 9 Sam drew 8 Pat has high card Pat: 2 Sam: 2

Pat drew 7 Sam drew 7 War Pat: 2 Sam: 2

Pat drew 5 Sam drew 6 Sam has high card Sam wins war of 4 cards Pat: 2 Sam: 6

…

Pat drew 10 Sam drew K Sam has high card Pat: 18 Sam: 24

Pat drew 2 Sam drew 2 War
Pat: 18 Sam: 24

Pat drew A Sam drew A War
Pat: 18 Sam: 24

Pat drew 2 Sam drew 5 Sam has high card Sam wins war of 6 cards Pat: 18 Sam: 30

Pat drew J Sam drew A Sam has high card Pat: 18 Sam: 32

Pat drew 10 Sam drew 3 Pat has high card Pat: 20 Sam: 32

Final Score Pat: 20 Sam: 32 Winner: Sam
```

Deck Shuffling

While seemingly simple, shuffling a deck is a somewhat complicated problem. Luckily, Python’s random library has a built-in shuffle algorithm. Feel free to read the documentation, but we have provided a simple wrapper function that will return to you a shuffled deck of cards.

```python
import random

# Name: shuffled_deck
# Purpose: will return a shuffled deck to the user
# Input:
# Output: a list representing a shuffled deck
def shuffled_deck():
    basic_deck = list(range(2, 15)) * 4
    random.shuffle(basic_deck)
    return basic_deck
```

Bonus

Instead of closing the program when the deck is empty, create a way for the user to play again.