

```

1 import random
2
3 suits = ('Hearts', 'Diamonds', 'Spades', 'Clubs')
4 ranks = ('Two', 'Three', 'Four', 'Five', 'Six', 'Seven', 'Eight', 'Nine', 'Ten', 'Jack', 'Queen', 'King', 'Ace')
5 values = {'Two':2, 'Three':3, 'Four':4, 'Five':5, 'Six':6, 'Seven':7, 'Eight':8, 'Nine':9, 'Ten':10, 'Jack':11,
6           'Queen':12, 'King':13, 'Ace':14}
7
8 class Card:
9
10     def __init__(self, suit, rank):
11         self.suit = suit
12         self.rank = rank
13         self.value = values[rank]
14
15     def __str__(self):
16         return self.rank + " of " + self.suit
17
18 class Deck:
19     def __init__(self):
20
21         self.all_cards = []
22
23         for suit in suits:
24             for rank in ranks:
25                 # creating a card object
26                 created_card = Card(suit,rank)
27                 self.all_cards.append(created_card)
28
29         # Shuffling the cards randomly:
30
31     def shuffle(self):
32         random.shuffle(self.all_cards)
33
34         # Grabbing one card from the list:
35
36     def deal_one(self):
37         return self.all_cards.pop()
38
39 class Player:
40
41     def __init__(self, name):
42         self.name = name
43         self.all_cards = []
44
45     def remove_one(self):
46         return self.all_cards.pop(0)
47
48     def add_cards(self, new_cards):
49         if type(new_cards) == type([]):
50             # List of multiple card objects
51             self.all_cards.extend(new_cards)
52         else:
53             # For a single card object
54             self.all_cards.append(new_cards)
55
56     def __str__(self):
57         return f"Player {self.name} has {len(self.all_cards)} cards."
58
59 #game Logic:
60
61 player_one = Player("One")
62 player_two = Player("Two")
63
64 new_deck = Deck()
65 new_deck.shuffle()
66
67 for x in range(26):
68     player_one.add_cards(new_deck.deal_one())
69     player_two.add_cards(new_deck.deal_one())
70
71 game_on = True
72
73 round_num = 0
74 while game_on:
75
76     round_num += 1
77     print(f"Round {round_num}")
78
79     # Check to see if a player is out of cards:
80     if len(player_one.all_cards) == 0:
81         print("Player One out of cards! Game Over")
82         print("Player Two Wins!")
83         game_on = False
84         break
85
86     if len(player_two.all_cards) == 0:
87         print("Player Two out of cards! Game Over")
88         print("Player One Wins!")
89         game_on = False
90         break
91
92     # Otherwise, the game is still on!
93     # Start a new round and reset current cards "on the table"
94
95     player_one_cards = []
96     player_one_cards.append(player_one.remove_one())
97
98     player_two_cards = []
99     player_two_cards.append(player_two.remove_one())
100
101     at_war = True
102     while at_war:
103
104         if player_one_cards[-1].value > player_two_cards[-1].value:
105
106             # Player One gets the cards
107             player_one.add_cards(player_one_cards)
108             player_one.add_cards(player_two_cards)
109
110             # No longer at "war" , time for next round
111             at_war = False
112
113         # Player Two Has higher Card
114         elif player_one_cards[-1].value < player_two_cards[-1].value:
115
116             # Player Two gets the cards
117             player_two.add_cards(player_one_cards)
118             player_two.add_cards(player_two_cards)
119
120             # No longer at "war" , time for next round
121             at_war = False
122
123         else:
124             print("WAR!")
125             # This occurs when the cards are equal.
126             # We'll grab another card each and continue the current war.
127             # First check to see if player has enough cards
128
129             # Check to see if a player is out of cards:
130             if len(player_one.all_cards) < 5:
131                 print("Player One unable to play war! Game Over at War")
132                 print("Player Two Wins! Player One Loses!")
133                 game_on = False
134                 break
135
136             elif len(player_two.all_cards) < 5:
137                 print("Player Two unable to play war! Game Over at War")
138                 print("Player One Wins! Player One Loses!")
139                 game_on = False
140                 break
141
142             # Otherwise, we're still at war, so we'll add the next cards
143
144             else:
145                 for num in range(5):
146                     player_one_cards.append(player_one.remove_one())
147                     player_two_cards.append(player_two.remove_one())

```