Author: Namzus Sakib, Student ID: 33361991, Tutor: Debabrata Mishra

FIT9136 S1 2023

In [1]:
```
import random
```

Task 1: Game Menu Function

In [2]:
```
def game_menu():
    """
    This is the game menu where the options will be displayed
    """
    # print("Welcome to card game, You have the following options in the game menu.")
    print("\nGame Menu:")
    print("1. Start Game")
    print("2. Pick a Card")
    print("3. Shuffle Deck")
    print("4. Show My Cards")
    print("5. Check Win/Lose")
    print("6. Exit")
    print(" ")

# This is for testing

# game_menu()
```

Task 2: Create Deck Function

In [3]:
```
def create_deck(deck, values, suits):
    """
    The deck is an empty list which is passed from the play_game function.
    The values are a constant list which is defined in the  play_game function.
    The suits is a list of elements that the player chooses to play with.
    """

    # This function creates the deck with combining every value with every suit elemer
    # A card is created for every value with every suit and then appended in the deck
    for suit in suits:
        for value in values:
            card = f"{value} of {suit}"
            deck.append(card)
    print("The deck has been created: ")
    print(" ")
    print(deck)

# This is for testing.

# deck = []
# values = ["2", "3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K", "A"]
# suits = ["♥", "♦", "♣", "♠"]

# create_deck(deck, values, suits)
```

Task 3: Shuffle Deck Fucntion

In [4]:
```
def shuffle_deck(deck, suits):
    """
    The deck is passed from the create deck function which is the list of cards.
    The suits is the suit type and the suit elements that the player chose to play wit
    """

    # Check if the deck is empty.
    if len(deck) == 0:
        print("The deck is empty!")
        return None

    # Get the required cards in the deck.
    a_index = "A of " + suits[0]
    q_index = "Q of " + suits[1]
    k_index = "K of " + suits[-1]

    # Shuffle the entire deck.
    random.shuffle(deck)

    # Remove the required cards from the deck and insert them in the appropriate place
    if a_index in deck:
        deck.pop(deck.index(a_index)) # here the A of first suit element is removed fr
        deck.insert(0, "A of " + suits[0]) # here the A of first suit element is inser
    if q_index in deck:
        deck.pop(deck.index(q_index)) # here the Q of second suit element is removed f
        deck.insert(round((len(deck)+1)/2), "Q of " + suits[1]) # here the Q of second
    if k_index in deck:
        deck.pop(deck.index(k_index)) # here the K of last suit element is removed fro
        deck.insert(len(deck), "K of " + suits[-1]) # here the K of last suit element


    # Display the shuffled deck with the cards in appropriate place to the player.
    print("Shuffled Deck: ")
    print(" ")
    print(deck)
    print(" ")
    print("The length of the deck is: ",len(deck)) # this shows the length of the deck
    print("The index of the Q of 2nd suit is: ",deck.index(q_index)) # This shows the

# This is for testing

# deck = ['4 of ♥', '5 of ♥', '6 of ♥', '7 of ♥', '8 of ♥', '9 of ♥', '10 of ♥', 'J of
# suits = ["♥", "♦", "♣", "♠"]

# shuffle_deck(deck, suits)
```

Task 4: Pick Card Function

In [5]:
```
def pick_card(deck, player_card, robot_card):
    """
    The deck is passed from the create deck function or the shuffle deck function whic
    The player_card is an empty list which was defined in the play_game function which
```

```
    The robot_card is an empty list which was defined in the play_game function which
    """

    # Check if the deck is empty.
    if len(deck) == 0:
        print("The deck is empty!")
        return None

    # Pick a random card from the deck for player.
    index_player = random.randint(0, len(deck) - 1) # here index_player get the random
    card_player = deck[index_player] # here the card of the index that was randomly pi
    player_card.append(card_player) # here the card is appended in the list player_car

    print("The player picked the card: ",card_player) # the card that is picked by the

    # Remove the picked card from the deck.
    deck.pop(index_player)

    # Select if the robot picks a card.
    robot_pick_chance = random.randint(0, 100) # here a random number is taken for the

    # Pick a random card from the deck for robot if the robot_pick_chance is over 70.
    if robot_pick_chance >= 70:
        index_robot = random.randint(0, len(deck) - 1) # here index_robot get the rand
        card_robot = deck[index_robot] # here the card of the index that was randomly
        robot_card.append(card_robot)# here the card is appended in the list robot_car
        deck.pop(index_robot) # here the picked card from the robot is removed from th



# This is for Testing

# deck = ['2 of ♥', '3 of ♥', '4 of ♥', '5 of ♥', '6 of ♥', '7 of ♥', '8 of ♥', '9 of
# player_card = []
# robot_card = []

# pick_card(deck, player_card, robot_card)
```

Task 5: Show Cards Function

In [6]:

```
def show_cards(player_cards):
    """
    The player_cards is the list of the card picked by the player is passed from the f
    """
    # Check if the player card list is empty.
    if len(player_cards) == 0:
        print("The player has not picked any card yet!")
        return None

    # This for loop print all the cards that the player picked.
    print("The players cards are: ")
    for card in player_cards:
        print(card)

# This is for testing

# player_cards = ['2 of ♥', '3 of ♥', '4 of ♥', '5 of ♥', '6 of ♥', '7 of ♥']
```

```
# show_cards(player_cards)
```

Task 6: Check Results Function

In [7]:

```
def check_result(player_cards, robot_cards, suits):
    """
    The player_cards is a list of cards that the player picked. This is passed from th
    The robot_cards is a list of cards that the robot picked. This is passed from the
    The suits is a list of suit elements that was passed from the play game function.
    """

    # Check if the player card is empty.
    if len(player_cards) == 0:
        print("The player has not picked any card yet!")
        return None

    state = 0 # this is to determine the state of the game. if the state is 1 game fir
    rule = 1 # initially we only check the rule number 1 only, if fails then go to the
    player_final_card = [] # this is a empty list that is needed for rule number 3 and
    robot_final_card = [] # this is a empty list that is needed for rule number 3 and
    player_wins = False # this denoted that the player has not won yet. If this is Tru
    robot_wins = False # this denoted that the robot has not won yet. If this is True
    position_2_suit = suits[1] # this store the suit element that was in position 2 wh
    player_score = 0 # this store the player score which is only needed for the rule r
    robot_score = 0 # this store the robot score which is only needed for the rule num
    player_average = 0 # this store the player average score which is only needed for
    robot_average = 0 # this store the robot average score which is only needed for th

    while state == 0: # this checks if the state of the game is set to continue or fir
        # Rule 1: same card for all the suits.
        if rule == 1:
            #c ode for rule 1
            # this code block checks every card and sets player_wins to true when if t
            for cards in player_cards:
                player_final_card.append(cards.split(" ")[0])
                if player_final_card.count(cards.split(" ")[0]) == len(suits):
                    player_wins = True

            # this code block checks every card and sets robot_wins to true when if th
            for cards in robot_cards:
                robot_final_card.append(cards.split(" ")[0])
                if robot_final_card.count(cards.split(" ")[0]) == len(suits):
                    robot_wins = True

            if robot_wins == True and player_wins == True: # here if both robot and pl
                rule = 2
            elif robot_wins == True and player_wins != True: # here it checks if the r
                print("By the rule number 1")
                print("The robot wins the game")
                state = 1
            elif robot_wins != True and player_wins == True: # here it checks if the p
                print("By the rule number 1")
                print("The player wins the game")
                state = 1

        # Rule 2: same card for 1 less than the elements in the suits.
```

```
        if rule == 2:
            # code for rule 2
            # this code block checks every card and sets player_wins to true when if t
            for cards in player_cards:
                player_final_card.append(cards.split(" ")[0])
                if player_final_card.count(cards.split(" ")[0]) == len(suits)-1:
                    player_wins = True

            # this code block checks every card and sets robot_wins to true when if th
            for cards in robot_cards:
                robot_final_card.append(cards.split(" ")[0])
                if robot_final_card.count(cards.split(" ")[0]) == len(suits)-1:
                    robot_wins = True

            if robot_wins == True and player_wins == True: # here if both robot and pl
                rule = 3
            elif robot_wins == True and player_wins != True: # here it checks if the r
                print("By the rule number 2")
                print("The robot wins the game")
                state = 1
            elif robot_wins != True and player_wins == True: # here it checks if the p
                print("By the rule number 2")
                print("The player wins the game")
                state = 1

        # Rule 3: Who holds the more cards that was in position 2 in the suits.
        if rule == 3:
            # code for rule 3
            # this block of code appends the suit elements of the position 2 of the su
            player_final_card = [card for card in player_cards if card.endswith(f" of
            robot_final_card = [card for card in robot_cards if card.endswith(f" of {p
            if len(player_final_card) > len(robot_final_card): # checks if the length
                player_wins = True
                print("By the rule number 3")
                print("The player wins the game")
                state = 1
            elif len(player_final_card) < len(robot_final_card): # checks if the lengt
                robot_wins = True
                print("By the rule number 3")
                print("The robot wins the game")
                state = 1
            elif len(player_final_card) == len(robot_final_card): # checks if the leng
                 rule = 4

        # Rule 4: Who holds the higher average of the card's value.
        if rule == 4:
            # code for rule 4
            # this is for getting the index where these letters are found.
            a_index = 'A'
            q_index = 'Q'
            j_index = 'J'
            k_index = 'K'

            player_final_card = [(card.split(" ")[0]) for card in player_cards] # here
            robot_final_card = [(card.split(" ")[0]) for card in robot_cards] # here t

            # here it checks if the value is digit then turns it into integer and adds
            for cards in player_final_card:
                if cards.isdigit():
                    player_score += int(cards)
```

```
                else:
                    if cards == a_index:
                        player_score = player_score + 1 # here if the card has a value
                    if cards == j_index:
                        player_score = player_score + 11 # here if the card has a valu
                    if cards == q_index:
                        player_score = player_score + 12 # here if the card has a valu
                    if cards == k_index:
                        player_score = player_score + 13 # here if the card has a valu

            # here it checks if the value is digit then turns it into integer and adds
            for cards in robot_final_card:
                if cards.isdigit():
                    robot_score += int(cards)
                else:
                    if cards == a_index:
                        robot_score = robot_score + 1 # here if the card has a value c
                    if cards == j_index:
                        robot_score = robot_score + 11 # here if the card has a value
                    if cards == q_index:
                        robot_score = robot_score + 12 # here if the card has a value
                    if cards == k_index:
                        robot_score = robot_score + 13 # here if the card has a value

            player_average = player_score / len(player_final_card) # here the average
            robot_average = robot_score / len(robot_final_card) # here the average sco

            if player_average > robot_average: # if the player's average is more than
                print("By the rule number 4")
                print("The player wins the game")
                state = 1
            elif player_average < robot_average: # if the player's average is less tha
                print("By the rule number 4")
                print("The robot wins the game")
                state = 1
            elif player_average == robot_average: # if the player's average is equal t
                print("The game is a TIE")
                state = 1

# This is for testing

# player_cards = ["A of ♥", "A of ♣", "Q of ♣", "10 of ♠", "Q of ♣","Q of ♣"]
# robot_cards = ["A of ♥", "A of ♣", "Q of ♣", "10 of ♠", "Q of ♣","Q of ♣"]
# suits = ["♥", "♦", "♣", "♠"]

# check_result(player_cards, robot_cards, suits)
```

Task 7: Play Game Function

In [8]:

```
def play_game():

    state = 0 # this is to determine the state of the game. if the state is 1 game fir
    deck = [] # this is an empty list which will be used to hold the cards of every va
    suits = [] # this is an empty list which will hold the suit elements of the select
    values = ["2", "3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K", "A"] # thes
    player_cards = [] # this is an empty list which will hold the cards that the playe
    robot_cards = [] # this is an empty list which will hold the cards that the robot
```

```
while state == 0:
    if len(player_cards)<6: # this condition is placed because a player can only p

        game_menu()

        first_suit = "0" # this is to hold the first value of the user input.
        second_suit = "0" # this is to hold the second value of the user input.

        user_input = input("Please Enter The Menu Number Here: ")

        if user_input.isdigit() == False: # this checks if the user input is alpha
            print("Please enter the valid menu number here: ")

        first_suit = user_input.split(" ")[0] # the user input is split and first

        # this block of code checks if there is a second value then it assigns it
        try:
            second_suit = user_input.split(" ")[1]
        except IndexError:
            second_suit = "0"

        if first_suit == "1":
            if (len(deck)>=1): # this checks if the player has already started a g
                print("You have already started a game. Continue with the game or

            if (len(deck) == 0) and (second_suit == "1" or second_suit == "0"): #
                suits_full = ["♥", "♦", "♣", "♠"]
                suit_element = input("Select Number of Elements you want to play w

                # this block of code appends the selected number of suit element i
                if suit_element.isdigit():
                    suit_index = 0
                    if int(suit_element)>=2 and int(suit_element)<=len(suits_full)
                        while suit_index != int(suit_element):
                            suits.append(suits_full[suit_index])
                            suit_index += 1
                        create_deck(deck, values, suits)

            if (len(deck) == 0) and second_suit == "2":
                suits_full = ["😀", "😈", "😵", "🤢", "😨"]
                suit_element = input("Select Number of Elements you want to play w

                # this block of code appends the selected number of suit element i
                if suit_element.isdigit():
                    suit_index = 0
                    if int(suit_element)>=2 and int(suit_element)<=len(suits_full)
                        while suit_index != int(suit_element):
                            suits.append(suits_full[suit_index])
                            suit_index += 1
                        create_deck(deck, values, suits)

            if (len(deck) == 0) and second_suit == "3":
                suits_full = ["🤡", "👹", "👺", "👻", "👽", "👾", "🤖"]
                suit_element = input("Select Number of Elements you want to play w

                # this block of code appends the selected number of suit element i
                if suit_element.isdigit():
                    suit_index = 0
                    if int(suit_element)>=2 and int(suit_element)<=len(suits_full)
```

```
                            while suit_index != int(suit_element):
                                suits.append(suits_full[suit_index])
                                suit_index += 1
                            create_deck(deck, values, suits)

            # this checks if the user input is 2 calls pick_card function.
            elif first_suit == "2":
                if len(deck) == 0: # this checks if the length of the deck is 0 then t
                    print("The game has not started yet. Please start the game")
                pick_card(deck, player_cards, robot_cards)

            # this checks if the user input is 3 calls shuffle_deck function.
            elif first_suit == "3":
                if len(deck) == 0: # this checks if the length of the deck is 0 then t
                    print("The game has not started yet. Please start the game")
                shuffle_deck(deck, suits)

            # this checks if the user input is 4 calls show_cards function.
            elif first_suit == "4":
                if len(deck) == 0: # this checks if the length of the deck is 0 then t
                    print("The game has not started yet. Please start the game")
                show_cards(player_cards)

            # this checks if the user input is 5 calls check_result function.
            elif first_suit == "5":
                if len(deck) == 0: # this checks if the length of the deck is 0 then t
                    print("The game has not started yet. Please start the game")
                check_result(player_cards, robot_cards, suits)

            # this checks if the user input is 6, this ends the game.
            elif first_suit == "6":
                print("The game has ended successfully")
                state = 1
        else:
            check_result(player_cards, robot_cards, suits)
            state = 1
```

In [9]:
```
play_game()
```

```
Game Menu:
1. Start Game
2. Pick a Card
3. Shuffle Deck
4. Show My Cards
5. Check Win/Lose
6. Exit

Please Enter The Menu Number Here: 6
The game has ended successfully
```

In [ ]:

In [ ]:

In [ ]:

In [ ]: