

Multiplayer Black Jack game

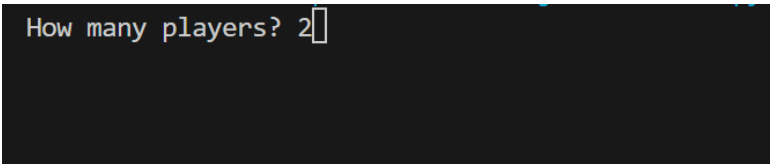
Here are the key features of my Black Jack game.

1. Multiplayer Support: The game will prompt you to specify the number of players.
2. Text-Based: The game is text-based. Players will interact with the terminal.
3. The game will follow standard black-jack rules.

Design and Implementation

1. Design:
 - a. Game will prompt user to input the number of players
 - b. The game will dynamically create instances of the 'Player' class
 - c. The game will use 1 standard deck containing 52 cards
 - d. The game will randomly select a card from the deck when dealing
 - e. The game will deal out cards until each player has 2 including the dealer
 - f. The game will show all the players cards along with the total
 - g. Each player will be prompted to Hit or Stay until the total > 21 or they type Stay
 - h. The dealer will continue to Hit until total is >= 17
 - i. At the end of the game, Results will be printed out

2. Implementation:
 - a. Number of players input will be stored in a variable num_players



```
How many players? 2
```

- b. A list of 'Player' objects will be created using a for loop with num_players as the stop value.
- c. The 'Player' class
 - i. Attributes:
 1. Name: String
 2. Cards: Dictionary
 3. is_bust: Boolean
 4. is_dealer: Boolean
 - ii. Method:
 1. Sum_of_cards: returns the some of the values in Cards
- d. The Deck will be a dictionary data type representing a 52 card deck.
 - i. The key will be the card ex: "A of spades"
 - ii. The value will be the value of the card ex: "A of spades": 11
- e. The deal_cards() function will take in an argument of 'Player' object

- i. `get_card()` function will be called and return value will be assigned to `player.card` attribute
 - ii. `get_card()`
 - 1. initialize an empty dictionary called `card`
 - 2. randomly select a card from `card_deck` using `random.choice()`
 - 3. add card to card dictionary
 - 4. remove card from `card_deck`
 - 5. return card
- f. The `show_cards()` function
 - i. Loop through players list
 - 1. Print player cards (Keys only)
 - 2. Print player card total
- g. After the initial 2 cards are dealt. Game will loop through player list and prompt 'Hit' or 'Stay' until the player types 'Stay' or the player busts(total > 21)

```
Dealer has: ['Q of spades', 'J of hearts']
Total: 20

Player 1 has: ['2 of diamonds', '9 of clubs']
Total: 11

Player 2 has: ['2 of clubs', '2 of spades']
Total: 4

Player 1 hit or stay? : 
```

- h. Dealer will have the last action
 - i. Dealer will automatically hit(`get_card`) until total > 17
- i. The game will end
 - i. `Show_winner()` function will print the results of the game

```
Dealer has: ['Q of spades', 'J of hearts']
Total: 20

Player 1 has: ['2 of diamonds', '9 of clubs', 'Q of diamonds']
Total: 21

Player 2 has: ['2 of clubs', '2 of spades', 'Q of clubs', 'J of clubs']
Total: 24

-----
      Game Results
-----

Player 1 wins!
Player 2 busted. Dealer wins!
PS C:\Users\Brian\Desktop\NucampFolder\Python\1-Fundamentals\portfolio> 
```

3. Conclusion

This game was a great challenge and gave me the opportunity to apply what I've learned in the course. One of the features I like about the game is the ability to create multiple players. This added a little more complexity to the code but was a great experience to figure it out and actually apply to the game.

One of the challenges I faced when creating the game was trying to figure out the best way to implement game logic for handling "Soft Cards" which is a hand that contains an Ace that can be counted as either 1 or 11. Although I managed to get it to work, I feel there is a more efficient way to write the code to reduce the amount of conditional statements used.

In hindsight, I think creating a separate class for 'Dealer' would have been a better approach to the code. The function to deal cards would be a method in the Dealer class. Also the deck of cards could be a class on it's own with methods to shuffle and assign cards.

A feature that could be added in the future would be the ability to actually place a bet and have multiple deck of cards.