SECTION 11: MILESTONE PROJECT - 2 hours 18 minutes, 12 sections

- **6/12 Game Logic - Part Two**
  - -> programming game logic
  - -> the player class, desk class and card class
  - -> then suits / ranks / values defined in tuples

  - **-> in .ipynb file**
    - **for the game setup**
      - she's defined two players (instances of the class)
      - then made a new deck, then shuffled it <- this replaces the list with the shuffled version (it doesn't define a new one)
      - -> then she's used a for loop
        - this adds cards to the empty starting hands of the players
        - -> it's going up o 26 because there are two players and 52 cards in the entire deck
        - -> **running tests for the output as she goes on**

    - **-> she has defined a boolean -> the game is on (True)**
      - -> then a while loop, running while the game is on
      - **-> under the while loop**
        - round_num += 1
        - -> then printed the round_num

        - **-> then she is coding the different case scenarios**
          - the one where player 1 wins is the one where the length of the array which stores player 2's cards is empty and vice versa
            - this is done with booleans
            - -> **she is also adding a break loop because the entire thing is indented under a while loop (while the game is on, and it's initialised with this boolean as being on)**

        - **then she is starting the player cards**
          - -> player_one_cards = []
          - -> player_two_cards = []
          - -> both the players start off with no cards
          - -> she is adding cards to the empty list which store the cards / hands of each player -> this is done with the append method
          - -> to start a new round, she has used the .remove_one() term -> which pops a card from the array storing their hands

        - **so the thought process is**
          - -> she sets up players and decks
          - -> prints out round numbers
          - -> checks to see if a new round is needed or if the game has been won yet
          - -> removes a card from each players' hand
          - -> then we need to check the different case scenarios -> depending on what cards are removed from the hands of the players