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

- **2/12 Card Class**
    - -> the card class and global variables which can be used
    - -> the suit of the card, rank, value
    - -> a deck class (52 cards, generating a deck)
    - -> then the player

    - **-> in the .ipynb file**
        - **creates card class**
            - class Card:
                - def __init__(self, suit, rank) <- we initialise the class
                    - -> the value is automatically given (we write code to do this later)
                    - -> self.suit = suit
                    - -> self.rank = rank
                - def __str__(self): <- then we are defining methods to use on the card
                    - return self.rank + " of " + self.suit

        - **then create instances of the class to see if the code is working or not**
            - two_hearts = Card("Hearts", "Two")
            - -> then she's changed the class to include an __str__(self) section
            - -> then printing two_hearts returns information about it
            - -> she's then using a dictionary of values to convert "Two" into 2
                - i.e 'Two':2 <- 'Two' is a key in the dictionary

        - **you can then use values[two_hearts.rank]**
            - -> self.value = values[rank]
            - -> the values have to be entered in a certain form

    - three_of_clubs = Card("Clubs", "Three")
    - **then three_of_clubs. <- enter shift tab and you can see the different methods which can be ran**

    - **-> then she compares two cards**
        - **-> this can be done using a < statement**
        - -> card classes can represent global variables or an entire imported library
            - random shuffles

    - **-> she then creates two more objects**
        - -> lists of all the suits and ranks
        - -> tuples
        - **-> there is one tuple containing the suits (...,....,...,...)**
        - -> and then another for ranks
        - -> a dictionary for values (to convert 'Two' into 2 e.g with a key)

    - -> the suit and rank
    - -> and then the value lookup