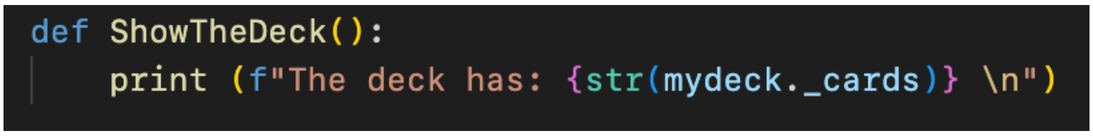
Step 2::
SHUFFLING THE DECK

1: Remove the print (self.cards) statement om the Deck's __init__ function - we know now that the cards exist - and do not need the whole deck printing out each time the program is run.

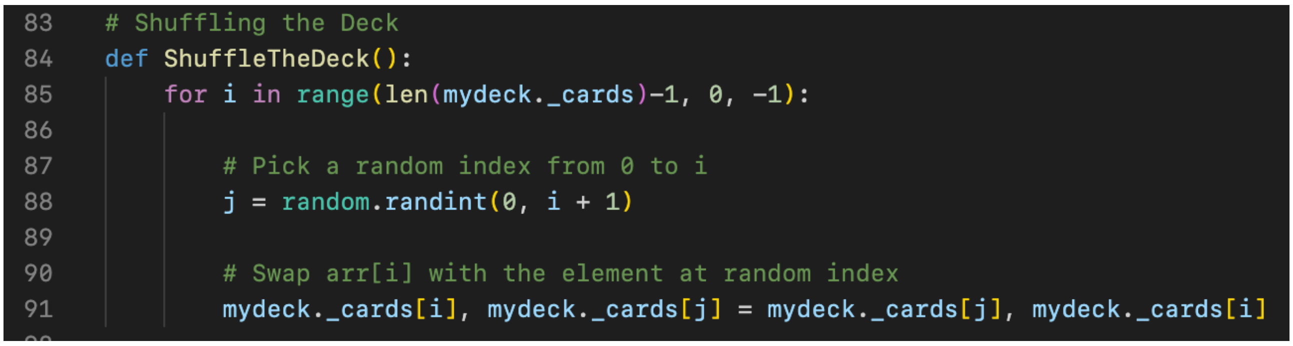
2: We will now instead create a function to show the cards before and after they have been shuffled.



```python
def ShowTheDeck():
    print (f"The deck has: {str(mydeck._cards)} \n")
```

This function shows all of the cards in the deck. The reason we have added this as a separate function is so that it we can show the cards more than once, to check whether our shuffle function has worked.

3: We Will now create a function that takes all of the cards in the deck and shuffles them.



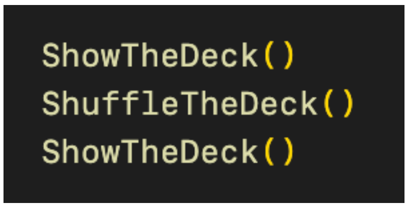
```python
# Shuffling the Deck
def ShuffleTheDeck():
    for i in range(len(mydeck._cards)-1, 0, -1):

        # Pick a random index from 0 to i
        j = random.randint(0, i + 1)

        # Swap arr[i] with the element at random index
        mydeck._cards[i], mydeck._cards[j] = mydeck._cards[j], mydeck._cards[i]
```

This code is the Fisher-Yates shuffle algorithm takes the length of range of the deck of cards (52), writes down every value as a new list (i), and works its way down by incrementing the size of the list down in increments of -1.

J is a random integer between 0 and 1 more than the current length of the deck of cards. The values picked are what gets removed from the (i) list and appended to the (j) list

Once the lists have been worked through, the (j) list overwrites the (I) list, replacing the once ordered deck with a new, randomised version.

4: Time to test it!
We will use the ShowTheDeck() function to show the deck - then, we will shuffle, and then show the deck again



```python
ShowTheDeck()
ShuffleTheDeck()
ShowTheDeck()
```