Step 3
MOVING CARDS FROM THE DECK TO THE HAND

1:
Create a hand object = we will give the hand an empty list to move cards into, and its own value, which will be the total value of the cards moved into the hand. We will do this using the card values we created earlier

Note that we are using self to refer to the hands own values again

```
 99  class Hand:
100      def __init__(self, handvalue): #tell it to have self referential hand value
101          self.cards = []
102          self.handvalue = handvalue
```

2:
Then, we will create an instance of the hand - remember we need to add the initial value of the hand when it is initialised - this is always 0 because there are no cards in the hand yet!

```
106  myhand = Hand(0)
```

3: Lets deal a card = we will do this using a method.
We create a taken card variable to move a card into temporarily - this will allow us to pull the cards value and add it to the hand before moving the card itself into the hand.
We will first pop the card from the deck, using position 0 - this is always the first position in the list, and so is the first position in the deck. Lastly, the card we popped is now appended to the hand.

```
def MyDeal(): #create a method to deal the card
    takencard = mydeck._cards.pop (0) #pop the card from the list so we can hold it in memory
    to pull the value
    myhand.handvalue += takencard._value # increase the value of the hand by the value of the
    card
    myhand.cards.append(takencard)
```

3:
Now we create a method to show what is in the hand
We will use f strings to show what cards are in the hand, then how much the hand is worth by calling the variables we created for the hand objects earlier.

```
def ShowMyHand():
    print (f"I have {myhand.cards} in my hand")
    print (f"my hand is now worth {myhand.handvalue}") #show hand value
```

4:
Let's test it!
Using the methods we have created so far, let's create a for loop that deals a card and shows our hand twice over

```
for i in (1,2):
    MyDeal() # deal a card
    ShowMyHand()
```