**Objective**

The objective of this lab is to create the beginnings for a python-based card game. We DO NOT expect a fully functioning card game. What we do expect is that you create a main function and various functions that will accomplish the following goals:

* Build a single-dimension array to keep track of the location of every card
* DO NOT move cards around... Just use the array to keep track of where each card is
* All card data is really integers - **Use other arrays to translate integers to suits, ranks, and player names**
* All cards will start in the DECK
* Write a function that translates a card number to a card name. HINT - look at the suitName and rankName arrays
* Write a function to assign a card to a given player
* Dealing a card involves picking a card number and assigning a new location to the corresponding element of cardLoc
* Write a function that displays the location of every card. (Early versions should show numeric values for the card number and location. Later versions can include string values for prettier output.)
* Write a function that prints the name of every card in a given hand

**Hints**

* Most people make this way too complicated
* You don't need any arrays I didn't already give you
* Do not use a two-dimensional array (unless you want to for the blackbelt challenge)
* Computer memory does *not* work like actual cards. You are not moving things around.

**Starter Code**

Please begin by copying the following code to your editor. You will not need to change my code at all, but you will need to add several new functions to make it work correctly.

""" cardGame.py

basic card game framework

keeps track of card locations for as many hands as needed

"""

from random import \*

NUMCARDS = 52

DECK = 0

PLAYER = 1

COMP = 2

cardLoc = [0] \* NUMCARDS

suitName = ("hearts", "diamonds", "spades", "clubs")

rankName = ("Ace", "Two", "Three", "Four", "Five", "Six", "Seven",

"Eight", "Nine", "Ten", "Jack", "Queen", "King")

playerName = ("deck", "player", "computer")

def main():

clearDeck()

for i in range(5):

assignCard(PLAYER)

assignCard(COMP)

showDeck()

showHand(PLAYER)

showHand(COMP)

**Sample output**

Your program output should look something like this. (It's perfectly fine if things don't line up perfectly. It's the general structure I'm looking for.)

Location of all cards

# card location

0 Ace of hearts deck

1 Two of hearts computer

2 Three of hearts deck

3 Four of hearts deck

4 Five of hearts deck

5 Six of hearts player

6 Seven of hearts deck

7 Eight of hearts player

8 Nine of hearts computer

9 Ten of hearts deck

10 Jack of hearts deck

11 Queen of hearts deck

12 King of hearts deck

13 Ace of diamonds deck

14 Two of diamonds deck

15 Three of diamonds deck

16 Four of diamonds deck

17 Five of diamonds deck

18 Six of diamonds deck

19 Seven of diamonds deck

20 Eight of diamonds deck

21 Nine of diamonds deck

22 Ten of diamonds computer

23 Jack of diamonds deck

24 Queen of diamonds deck

25 King of diamonds deck

26 Ace of spades deck

27 Two of spades deck

28 Three of spades deck

29 Four of spades deck

30 Five of spades deck

31 Six of spades deck

32 Seven of spades player

33 Eight of spades deck

34 Nine of spades deck

35 Ten of spades deck

36 Jack of spades deck

37 Queen of spades deck

38 King of spades deck

39 Ace of clubs deck

40 Two of clubs deck

41 Three of clubs deck

42 Four of clubs deck

43 Five of clubs deck

44 Six of clubs deck

45 Seven of clubs deck

46 Eight of clubs deck

47 Nine of clubs computer

48 Ten of clubs computer

49 Jack of clubs player

50 Queen of clubs player

51 King of clubs deck

Displaying player hand:

Six of hearts

Eight of hearts

Seven of spades

Jack of clubs

Queen of clubs

Displaying computer hand:

Two of hearts

Nine of hearts

Ten of diamonds

Nine of clubs

Ten of clubs

**Blackbelt**

Use what you have created and extend it to actually implement a working card game. War is pretty easy, as is BlackJack. If you're feeling more ambitious, you might try something like "Go Fish." Poker is easy to write, but the scorekeeping can be tricky.

|  |  |
| --- | --- |
| Rubric | Points |
| Code: main() is relatively unchanged from Assignment | 2 |
| Code: clearDeck initializes CardLoc to DECK | 2 |
| Code: assignCard calls randrange() or randint() | 2 |
| Code: assignCard checks If cardLoc is DECK | 2 |
| Code: assignCard has location as a formal parameter | 2 |
| Code: assignCard sets random index to location | 2 |
| Code: showDeck displays heading similar to "Location of all cards" | 2 |
| Code: showDeck display column headings #, card and location | 2 |
| Code: showDeck uses extended for in range() syntax | 2 |
| Code: showDeck displays index 0 to 51 | 2 |
| Code: showDeck displays index | 2 |
| Code: showDeck displays suitName[index/13], or equivalent | 3 |
| Code: showDeck displays rankName[index%13], or equivalent | 3 |
| Code: showDeck displays playerName[index], or equivalent | 2 |
| Code: showHand has player is a formal parameter | 2 |
| Code: showHand loops through cardLoc[] looking for cardLoc[i] = player | 2 |
| Code: showHand displays headingsimilar to "Player "+playerName[cardLoc[i]]+" hand:" | 2 |
| Code: showHand displays suitName[index/13], or equivalent | 3 |
| Code: showHand displays rankName[index%13], or equivalent | 3 |
| Execution: First section has title similar to "Location of all cards" | 2 |
| Execution: First section has column headings | 2 |
| Execution: Second section has title similar to "Displaying player hand:" | 2 |
| Execution: cards displayed correspond to "player" in show deck. | 2 |
| Execution: Third section has title similar to "Displaying computer hand:" | 2 |
| Execution: cards displayed correspond to "computer" in show deck. | 2 |
| Execution: running the program again changes the player and computer hands | 2 |
| Blackbelt: Working Card Game as a separate submission | 6 |
|  |  |
| Total | 56 |

""" cardGame.py

basic card game framework

keeps track of card locations for as many hands as needed

"""

from random import \*

NUMCARDS = 52

DECK = 0

PLAYER = 1

COMP = 2

card = [0] \* NUMCARDS

cardLoc = [0] \* NUMCARDS

suitName = ("hearts", "diamonds", "spades", "clubs")

rankName = ("Ace", "Two", "Three", "Four", "Five", "Six", "Seven",

"Eight", "Nine", "Ten", "Jack", "Queen", "King")

playerName = ("deck", "player", "computer")

def main():

clearDeck()

for i in range(5):

assignCard(PLAYER)

assignCard(COMP)

showDeck()

showHand(PLAYER)

showHand(COMP)

def clearDeck():

for num in range(NUMCARDS):

card[num] = str(rankName[num%13])+" of "+str(suitName[num/13])

cardLoc[num] = playerName[0]

def showDeck():

print "# " + "card "+" location"

for num in range(NUMCARDS):

print num, card[num],cardLoc[num]

def assignCard(hand):

rand\_num = randrange(0, NUMCARDS)

cardLoc[rand\_num] = playerName[hand]

def showHand(hand):

print "Displaying ", playerName[hand],"hand:"

for num in range(NUMCARDS):

if cardLoc[num] == playerName[hand]:

print card[num]

if \_\_name\_\_ == "\_\_main\_\_":

main()