Project 10 CS 261

Save your program for this project as `<emailID>_project10.py` *where* `<emailID>` *is the part of your Hampden-Sydney e-mail address before the @ symbol. When you are finished, e-mail your program to* `blins@hsc.edu`. *Your solution is due by noon on Friday, November 15.*

Deck of Cards

A regular deck of playing cards has 52 cards. There are 4 suits: clubs ♣, diamonds ♢, hearts ♡, and spades ♠, and each suit has thirteen ranks: 2 through 10, plus jacks, queens, kings, and aces. In this project, you will create two classes for working with a deck of cards.

1. Create a class called `Card`. Card objects should have two attributes, one for the suit and one for the rank. The `__init__` method should input two strings, one for the suit (`'C'`, `'D'`, `'H'`, or `'S'`) and the other for the rank.

2. Define the `__str__` method for the `Card` class. It should return a string with the card object's suit & rank in the following format. Use one of these characters (♣, ♢, ♡, ♠) for the suit followed by the letter or number corresponding to the rank. So, the jack of hearts would be ♡J, and the 10 of diamonds would be ♢10.

3. Create a class called `Deck`. Deck objects should have a `card_list` attribute that contains all 52 cards in the deck. The constructor function should initialize all 52 cards as instances of the `Card` class.

4. Add a method to the `Deck` class called `.shuffle()`. When you define the `shuffle()` method, it should only take `self` as an argument. It should use the `random.shuffle` function to shuffle the `card_list` of a deck object. You'll need to import the `random` module to do this.

5. Add a method called `.deal_cards(n)` to the `Deck` class. This method should pop n cards off of the `card_list` and then return a list containing those n cards.

6. Add a method called `.replace_card(card)`. This method should place a card back into the `card_list` of a deck object. It should also have an optional argument `bottom` which is `False` by default. If it is set to `True`, then the card should be placed on the bottom of the deck. Since the list methods `.pop()` and `.append()` remove and replace elements at the end of a list, you can think of the end of `card_list` as the "top" of the deck, and the front as the "bottom". Hint: you can use the `.insert(0, elem)` method to add an element to the front of a list.

7. To finish the project, create a deck object and use it to simulate dealing a hand of 5 cards. Check whether every card in the hand has the same suit (i.e., check if the hand would be a flush in poker). Then return the cards to the deck and shuffle it. Simulate this 10,000 times. How many of the hands in your simulation were flushes?