A PLAYING CARD GAME

A simple hi-lo card game where the user is given a card and they have to say if the next card will be higher or lower than it. I created classes to represent a card and a deck of cards, and these classes can be reused in other card games. We start with a class for a playing card. The data associated with a card consists of its value (2 through 14) and its suit. The Card class below has only one method, __str__. This is a special method that, among other things, tells the print function how to print a Card object.

```
class Card:
    def __init__(self, value, suit):
        self.value = value
        self.suit = suit
    def __str__(self):
        names = ['Jack', 'Queen', 'King', 'Ace']
        if self.value <= 10:
            return '{} of {}'.format(self.value, self.suit)
        else:
            return '{} of {}'.format(names[self.value - 11], self.suit)
```

Next we have a class to represent a group of cards. Its data consists of a list of Card objects. It has a number of methods: nextCard which removes the first card from the list and returns it; hasCard which returns True or False depending on if there are any cards left in the list; size, which returns how many cards are in the list; and shuffle, which shuffles the list.

```
import random
class Card_group:
    def __init__(self, cards=[]):
        self.cards = cards

    def nextCard(self):
        return self.cards.pop(0)

    def hasCard(self):
        return len(self.cards) > 0

    def size(self):
        return len(self.cards)

    def shuffle(self):
        random.shuffle(self.cards)
```

We have one more class Standard_deck, which inherits from Card_group. The idea here is that Card_group represents an arbitrary group of cards, and Standard_deck represents a specific group of cards, namely the standard deck of 52 cards used in most card games.

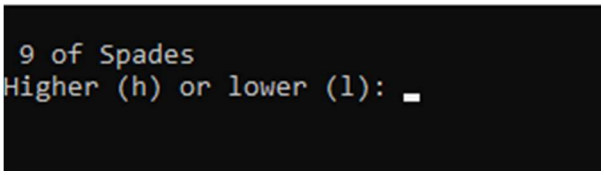
```
class Standard_deck(Card_group):
    def __init__(self):
        self.cards = []
        for s in ['Hearts', 'Diamonds', 'Clubs', 'Spades']:
            for v in range(2,15):
                self.cards.append(Card(v, s))
```

Here is the hi-low program that uses the classes we have developed here. One way to think of what we have done with the classes is that we have built up a miniature card programming language, where we can think about how a card game works and not have to worry about exactly how cards are shuffled or dealt or whatever, since that is wrapped up into the classes. For the hi-low game, we get a new deck of cards, shuffle it, and then deal out the cards one at a time. When we run out of cards, we get a new deck and shuffle it. A nice feature of this game is that it deals out all 52cards of a deck, so a player can use their memory to help them play the game.

```
deck = Standard_deck()
deck.shuffle()
new_card = deck.nextCard()
print('\n', new_card)
choice = input("Higher (h) or lower (l): ")
streak = 0
while (choice=='h' or choice=='l'):
    if not deck.hasCard():
        deck = Standard_deck()
        deck.shuffle()
    old_card = new_card
    new_card = deck.nextCard()
    if (choice.lower()=='h' and new_card.value>old_card.value or choice.lower()=='l' and new_card.value<old_card.value):
        streak = streak + 1
        print("Right! That's", streak, "in a row!")
    elif (choice.lower()=='h' and new_card.value<old_card.value or choice.lower()=='l' and new_card.value>old_card.value):
        streak = 0
        print('Wrong.')
    else:
        print('Push.')
    print('\n', new_card)
    choice = input("Higher (h) or lower (l): ")
```

Running the program:

1. Once you have downloaded the project then;
2. From the project directory navigate into the dist>main and double click on main.exe. A simple console window will appear as shown below;



Hurray! Enjoy the game!