# Zeru-Zhou-project10(1)

March 29, 2022

## 1 Project 10 – Zeru Zhou

**TA Help:** NA

**Collaboration:** NA

- Get help from dr. ward's videos

### 1.1 Question 1

[1]:
```
class Card:

    _value_dict = {"2": 2, "3": 3, "4": 4, "5": 5, "6": 6, "7": 7, "8":8, "9":
↪9, "10": 10, "j": 11, "q": 12, "k": 13, "a": 14}
    def __init__(self, number, suit):
        if str(number).lower() not in [str(num) for num in range(2, 11)] +␣
↪list("jqka"):
            raise Exception("Number wasn't 2-10 or J, Q, K, or A.")
        else:
            self.number = str(number).lower()
        if suit.lower() not in ["clubs", "hearts", "diamonds", "spades"]:
            raise Exception("Suit wasn't one of: clubs, hearts, spades, or␣
↪diamonds.")
        else:
            self.suit = suit.lower()

    def __str__(self):
        return(f'{self.number} of {self.suit.lower()}')

    def __repr__(self):
        return(f'Card(str({self.number}), "{self.suit}")')

    def __eq__(self, other):
        if self.number == other.number:
            return True
        else:
            return False
```

```
    def __lt__(self, other):
        if self._value_dict[self.number] < self._value_dict[other.number]:
            return True
        else:
            return False

    def __gt__(self, other):
        if self._value_dict[self.number] > self._value_dict[other.number]:
            return True
        else:
            return False

    def __hash__(self):
        return hash(self.number)
```

[2]:
```
class Deck:
    brand = "Bicycle"
    _suits = ["clubs", "hearts", "diamonds", "spades"]
    _numbers = [str(num) for num in range(2, 11)] + list("jqka")

    def __init__(self):
        self.cards = [Card(number, suit) for suit in self._suits for number in␣
 ↪self._numbers]

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

    def __getitem__(self, key):
        return self.cards[key]

    def __setitem__(self, key, value):
        self.cards[key] = value
```

[3]:
```
my_card = Card(5, "clubs")
print(my_card)
```

```
5 of clubs
```

[4]:
```
my_card
```

[4]:
```
Card(str(5), "clubs")
```

[5]:
```
my_deck = Deck()
print(my_deck)
```

```
<__main__.Deck object at 0x2b442d1a1ac0>
```

[10]:
```
class Deck:
    brand = "Bicycle"
    _suits = ["clubs", "hearts", "diamonds", "spades"]
    _numbers = [str(num) for num in range(2, 11)] + list("jqka")

    def __init__(self):
        self.cards = [Card(number, suit) for suit in self._suits for number in␣
 ↪self._numbers]

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

    def __getitem__(self, key):
        return self.cards[key]

    def __setitem__(self, key, value):
        self.cards[key] = value

    def __str__(self):
        return(f'A {self.brand.lower()} Deck')
```

[7]:
```
my_deck = Deck()
print(my_deck)
```

```
A bicycle Deck
```

[8]:
```
class Deck:
    brand = "Copag"
    _suits = ["clubs", "hearts", "diamonds", "spades"]
    _numbers = [str(num) for num in range(2, 11)] + list("jqka")

    def __init__(self):
        self.cards = [Card(number, suit) for suit in self._suits for number in␣
 ↪self._numbers]

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

    def __getitem__(self, key):
        return self.cards[key]

    def __setitem__(self, key, value):
        self.cards[key] = value

    def __str__(self):
        return(f'A {self.brand.lower()} Deck')
```

[9]:
```
my_deck = Deck()
print(my_deck)
```

```
A copag Deck
```

As above, modifications are made.

### 1.2 Question 2

[11]:
```
deck1 = Deck()
deck2 = Deck()
print(deck1)
print(deck2)
```

```
A bicycle Deck
A bicycle Deck
```

[12]:
```
Deck.brand = "Copag"
deck1 = Deck()
deck2 = Deck()
print(deck1)
print(deck2)
```

```
A copag Deck
A copag Deck
```

[13]:
```
deck1 = Deck()
deck2 = Deck()
deck1.brand = "Aviator"
Deck.brand = "Copag"
print(deck1)
print(deck2)
```

```
A aviator Deck
A copag Deck
```

The brand of deck1 is specifically modified to “Aviator”, and the basic brand for both 2 decks are changed to “Copag”.

### 1.3 Question 3

[9]:
```
class Player:

    def __init__(self, name, deck):
        self.name = name
        self.deck = deck
        self.hand = []

    def __str__(self):
```

```
        return(f"""
        {self.name}\n Top Cards:{self.deck[:5]}
        """)
```

[5]:
```
my_deck = Deck()
```

[6]:
```
player1 = Player("Roger Federer", my_deck)
print(player1)
```

```
        Roger Federer
 Top Cards:[Card(str(2), "clubs"), Card(str(3), "clubs"), Card(str(4), "clubs"),
Card(str(5), "clubs"), Card(str(6), "clubs")]
```

[7]:
```
import random
random.shuffle(my_deck)
player2 = Player("Eric Zhou", my_deck)
print(player2)
```

```
        Eric Zhou
 Top Cards:[Card(str(9), "hearts"), Card(str(k), "clubs"), Card(str(2),
"spades"), Card(str(6), "spades"), Card(str(5), "clubs")]
```

Class Player is created. The deck is shuffled.

### 1.4 Question 4

[3]:
```
class Player:

    def __init__(self, name, deck):
        self.name = name
        self.deck = deck
        self.hand = []

    def __str__(self):
        return(f"""
        {self.name}\n Top Cards:{self.deck[:5]}
        """)

    def draw(self):
        card = self.deck.cards.pop(0)
        self.hand.append(card)
```

[11]:
```
fresh_deck = Deck()
player1 = Player("Zeru Zhou", fresh_deck)
random.shuffle(fresh_deck)
player1.draw()
print(player1.hand)
```

```
[Card(str(q), "hearts")]
```

[12]:
```
player1.draw()
print(player1.hand)
```

```
[Card(str(q), "hearts"), Card(str(4), "diamonds")]
```

[13]:
```
player1.draw()
print(player1.hand)
```

```
[Card(str(q), "hearts"), Card(str(4), "diamonds"), Card(str(7), "hearts")]
```

Draw method is defined. Hand is also created.

### 1.5 Question 5

[7]:
```
class Player:

    def __init__(self, name, deck):
        self.name = name
        self.deck = deck
        self.hand = []

    def __str__(self):
        return(f"""
        {self.name}\n Top Cards:{self.deck[:5]}
        """)

    def draw(self):
        card = self.deck.cards.pop(0)
        self.hand.append(card)

    def has_set(self):
        count_hand = Counter(self.hand)
        for key, value in count_hand.items():
            if value >= 3:
                return True
        return False
```

[4]:
```
from collections import Counter
```

[8]:
```
import random
my_deck = Deck()
random.shuffle(my_deck)
player1 = Player("Zeru Zhou", my_deck)
for i in range(10):
    player1.draw()

player1.has_set()
```

[8]:
```
True
```

[10]:
```
print(player1.hand)
```

```
[Card(str(k), "hearts"), Card(str(3), "hearts"), Card(str(k), "spades"),
Card(str(6), "spades"), Card(str(6), "diamonds"), Card(str(9), "diamonds"),
Card(str(3), "clubs"), Card(str(8), "hearts"), Card(str(k), "diamonds"),
Card(str(4), "clubs")]
```

Here, we can see there are 3 “k” making that a set.

### 1.6 Pledge

By submitting this work I hereby pledge that this is my own, personal work. I’ve acknowledged in the designated place at the top of this file all sources that I used to complete said work, including but not limited to: online resources, books, and electronic communications. I’ve noted all collaboration with fellow students and/or TA’s. I did not copy or plagiarize another’s work.

> As a Boilermaker pursuing academic excellence, I pledge to be honest and true in all that I do. Accountable together – We are Purdue.