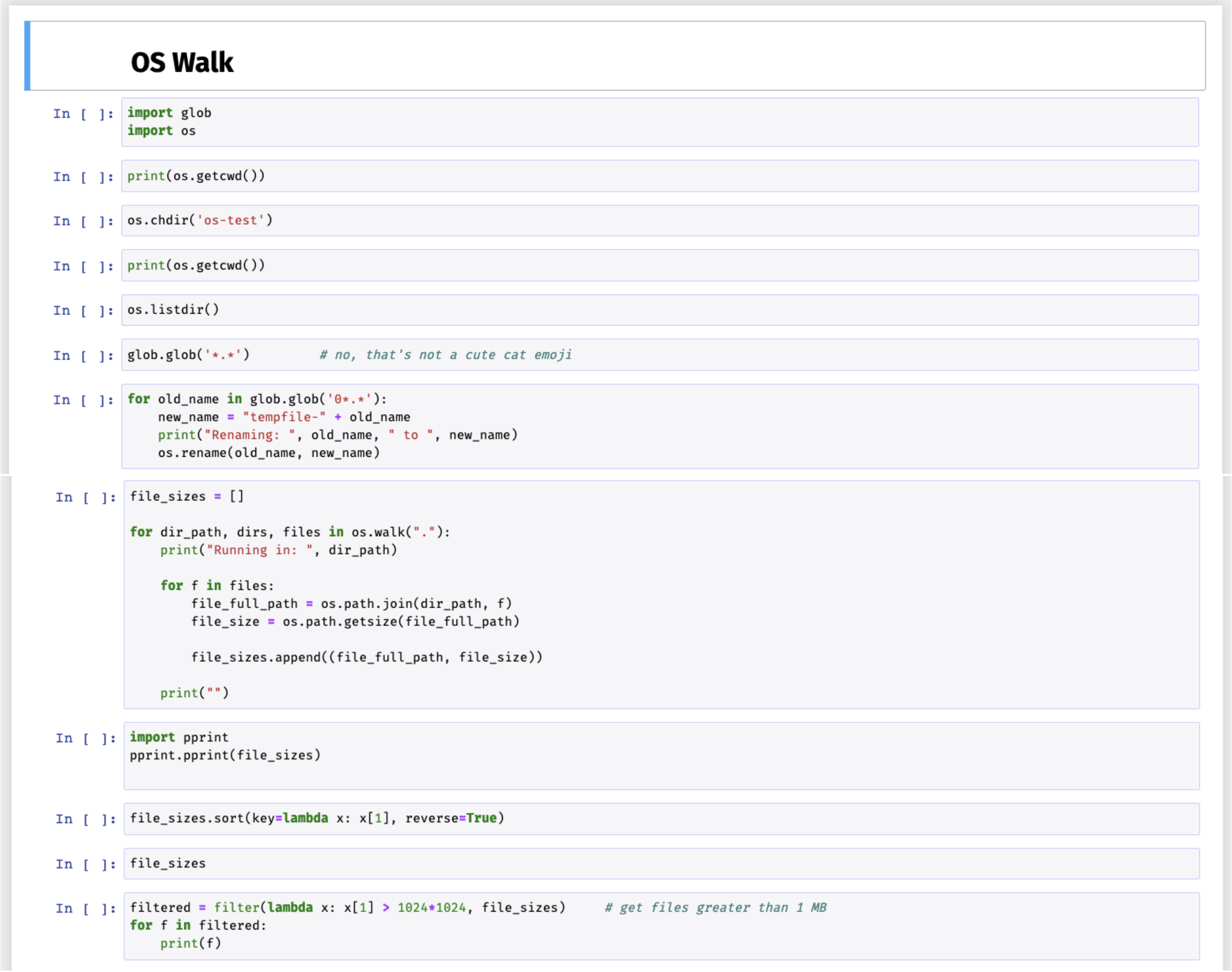
CS218 - Data Structures
FAST NUCES Peshawar Campus
Dr. Nauman (recluze.net)

November 19, 2019

1 Glob, OS Walk, Deck of Cards

Raster images of the notebook 19-misc.



OS Walk

```
In [ ]: import glob
        import os
```

```
In [ ]: print(os.getcwd())
```

```
In [ ]: os.chdir('os-test')
```

```
In [ ]: print(os.getcwd())
```

```
In [ ]: os.listdir()
```

```
In [ ]: glob.glob('*.*')          # no, that's not a cute cat emoji
```

```
In [ ]: for old_name in glob.glob('0*.*'):
            new_name = "tempfile-" + old_name
            print("Renaming: ", old_name, " to ", new_name)
            os.rename(old_name, new_name)
```

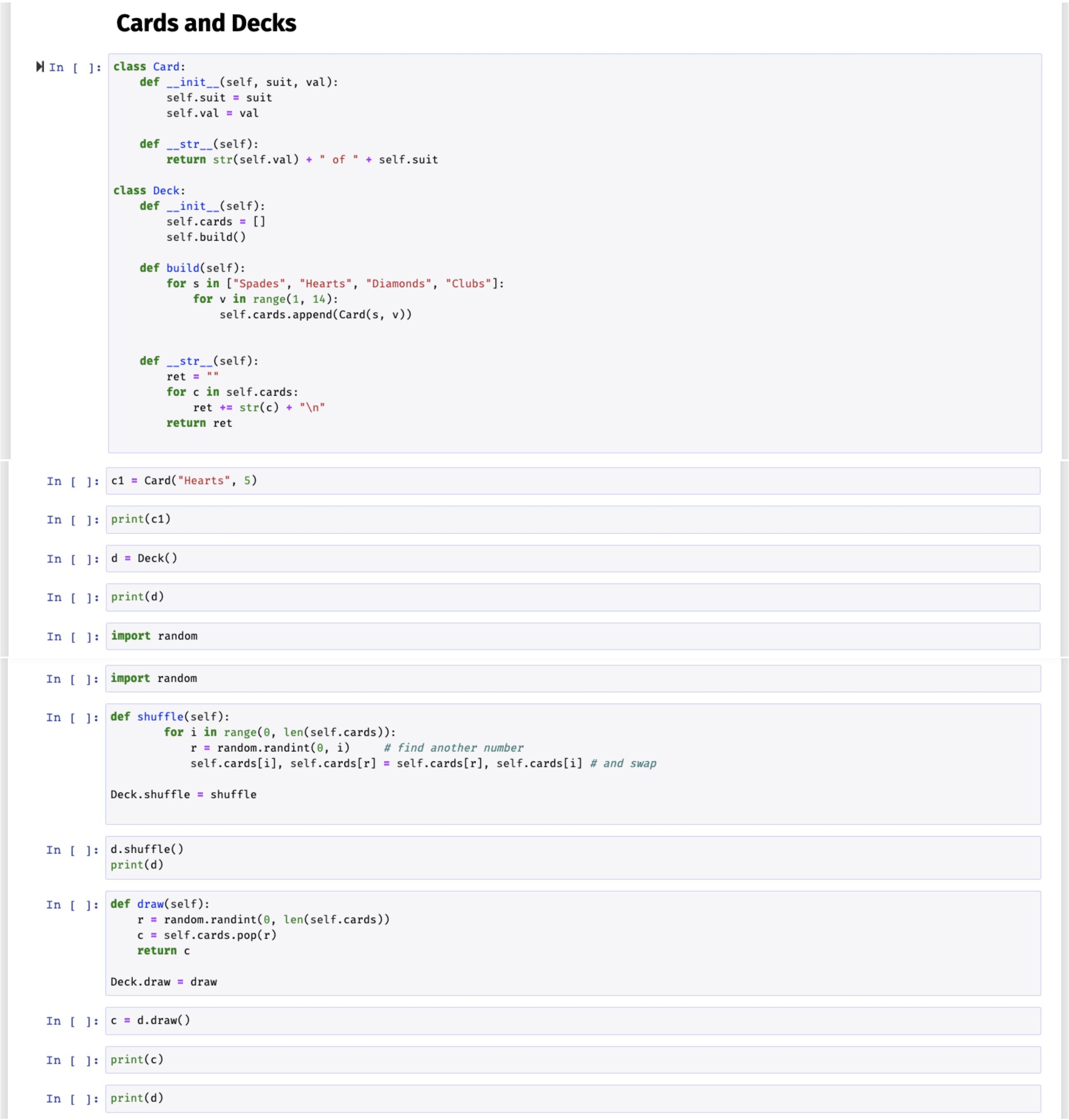
```
In [ ]: file_sizes = []

        for dir_path, dirs, files in os.walk("."):
            print("Running in: ", dir_path)

            for f in files:
                file_full_path = os.path.join(dir_path, f)
                file_size = os.path.getsize(file_full_path)

                file_sizes.append((file_full_path, file_size))

            print("")
```

```
In [ ]: import pprint
        pprint.pprint(file_sizes)
```

```
In [ ]: file_sizes.sort(key=lambda x: x[1], reverse=True)
```

```
In [ ]: file_sizes
```

```
In [ ]: filtered = filter(lambda x: x[1] > 1024*1024, file_sizes)     # get files greater than 1 MB
        for f in filtered:
            print(f)
```



Cards and Decks

In [ ]:
```
class Card:
    def __init__(self, suit, val):
        self.suit = suit
        self.val = val

    def __str__(self):
        return str(self.val) + " of " + self.suit

class Deck:
    def __init__(self):
        self.cards = []
        self.build()

    def build(self):
        for s in ["Spades", "Hearts", "Diamonds", "Clubs"]:
            for v in range(1, 14):
                self.cards.append(Card(s, v))


    def __str__(self):
        ret = ""
        for c in self.cards:
            ret += str(c) + "\n"
        return ret
```

In [ ]:
```
c1 = Card("Hearts", 5)
```

In [ ]:
```
print(c1)
```

In [ ]:
```
d = Deck()
```

In [ ]:
```
print(d)
```

In [ ]:
```
import random
```

In [ ]:
```
import random
```

In [ ]:
```
def shuffle(self):
        for i in range(0, len(self.cards)):
            r = random.randint(0, i)     # find another number
            self.cards[i], self.cards[r] = self.cards[r], self.cards[i] # and swap

Deck.shuffle = shuffle
```

In [ ]:
```
d.shuffle()
print(d)
```

In [ ]:
```
def draw(self):
    r = random.randint(0, len(self.cards))
    c = self.cards.pop(r)
    return c

Deck.draw = draw
```

In [ ]:
```
c = d.draw()
```

In [ ]:
```
print(c)
```

In [ ]:
```
print(d)
```