

✓ Essential Python for DA

- Object Oriented programming
- Read and Write file
- Context Manager
- Handle Error
- Pandas & Numpy(on demand)
- Request API

OOP

- is what we teach computer learn
- concept is like cookie cooker
- pum cookie cooker is class in python

```
class Dog:  
    pass
```

```
dog1 = Dog()  
dog2 = Dog()
```

```
dog2
```

```
<__main__.Dog at 0x7b418a8a3fb0>
```

```
class Dog:  
    def __init__(self, name, species,age):  
        self.name = name  
        self.species = species  
        self.age = age  
  
    def bark(self):  
        print(f"{self.name} is barking 'hong hong'!")  
  
    def hbd(self):  
        self.age += 1
```

```
dog1 = Dog("milo", "chiwawa",2)  
dog2 = Dog("david", "Husky",3)
```

```
dog1.name
```

```
'milo'
```

```
print(f"{dog1.name} is {dog1.species}")
```

```
milo is chiwawa
```

```
print(f"{dog2.name} is {dog2.species}")
```

```
david is Husky
```

```
dog1.bark()
```

```
miro is barking 'hong hong'!
```

```
print(dog1.age)
dog1.hbd()
print(dog1.age)
```

```
7
8
```

✓ OOP

- attributes: name, species, age
- method: function bark() hbd()

```
#home work OOP for ATM
```

```
##deposit,withdraw,check balance, OTP,order coffee, pay bill, internet , etc
```

```
class ATM:
    def __init__(self, username, bank_name, amount):
        self.username = username
        self.bank = bank_name
        self.amount = amount

    def __str__(self):    ##return string
        return f"{self.username} uses {self.bank} and has {self.amount} THB in an account"

    def check_balance(self):
        print(f"Blance: {self.amount} THB")

    def deposit(self, saving):
        self.amount += saving
        print(f"Deposit {saving} THB successfully!")
```

```
may = ATM("saruda may", "ttb", 100)
```

```
print(f"{may.username} opens a new bank account at {may.bank}")
```

```
saruda may opens a new bank account at ttb
```

```
may.check_balance()
```

```
Blance: 100 THB
```

```
may.deposit(200)
```

```
Deposit 200 THB successfully!
```

```
print(may)
```

```
saruda may uses ttb and has 300 THB in an account
```

✓ Super class

```
##base software
```

```
class Person:
```

```
    def __init__(self,name,age):
```

```
        self.name = name
```

```
        self.age = age
```

```
##extension / add feature
```

```
class Employee(Person):
```

```
    def __init__(self,name,age,company):
```

```
        super().__init__(name,age)
```

```
        self.company = company
```

```
    def greeting(self):
```

```
        print("Hello World!")
```

```
may = Person("may",39)
```

```
may.name, may.age
```

```
('may', 39)
```

```
person1= Person("may",39)
```

```
person2 = Employee("lisa",28,"google")
```

Try Except Block

```
try:
```

```
    result = 1/0
```

```
except:
```

```
    print("this is an error message!")
```

```
this is an error message!
```

✓ Read & Write Files

```
!ls
```

```
hotel.csv  sample_data
```

```
!cat hotel.csv
```

```
hotel_id,name,city,star_rating,price_per_night
101,Grand Plaza,New York,5,350
102,Ocean View,Miami,4,220
103,Mountain Retreat,Denver,4,185
104,The Urban Stay,Chicago,3,140
105,Sunset Resort,Los Angeles,5,410
106,Desert Inn,Phoenix,3,95
107,Heritage House,Boston,4,210
108,Lakeside Lodge,Seattle,3,130
109,Golden Gate Suites,San Francisco,5,380
110,Central Boutique,Austin,4,195
```

```
##read file
import csv
```

```
try:

    file = open("hotel.csv","r") # r=read,w=write,a=append

#csv.reader()
    data = csv.reader(file)

    for row in data:
        print(row)

    file.close()

except FileNotFoundError:
    print("file not found, please check filename again.")

finally:
    print("read file successfully")
```

```
['hotel_id', 'name', 'city', 'star_rating', 'price_per_night']
['101', 'Grand Plaza', 'New York', '5', '350']
['102', 'Ocean View', 'Miami', '4', '220']
['103', 'Mountain Retreat', 'Denver', '4', '185']
['104', 'The Urban Stay', 'Chicago', '3', '140']
['105', 'Sunset Resort', 'Los Angeles', '5', '410']
['106', 'Desert Inn', 'Phoenix', '3', '95']
['107', 'Heritage House', 'Boston', '4', '210']
['108', 'Lakeside Lodge', 'Seattle', '3', '130']
['109', 'Golden Gate Suites', 'San Francisco', '5', '380']
['110', 'Central Boutique', 'Austin', '4', '195']
read file successfully
```

```
#contex manager
read_data = []

with open("hotel.csv","r") as file:
    data = csv.reader(file)
    for row in data:
        read_data.append(row)
```

```
read_data
```

```
[['hotel_id', 'name', 'city', 'star_rating', 'price_per_night'],
 ['101', 'Grand Plaza', 'New York', '5', '350'],
```

```
[ '102', 'Ocean View', 'Miami', '4', '220'],
[ '103', 'Mountain Retreat', 'Denver', '4', '185'],
[ '104', 'The Urban Stay', 'Chicago', '3', '140'],
[ '105', 'Sunset Resort', 'Los Angeles', '5', '410'],
[ '106', 'Desert Inn', 'Phoenix', '3', '95'],
[ '107', 'Heritage House', 'Boston', '4', '210'],
[ '108', 'Lakeside Lodge', 'Seattle', '3', '130'],
[ '109', 'Golden Gate Suites', 'San Francisco', '5', '380'],
[ '110', 'Central Boutique', 'Austin', '4', '195']]
```

```
fruits = ["orange","banana","mango"]
fruits.append("durian")
```

```
print(fruits)
```

```
['orange', 'banana', 'mango', 'durian']
```

```
##R
#mean(data$price_per_night)
##python no average
## for loop to calculate avg. price per night

prices = []

for row in read_data[1: ]:
    prices.append(int(row[4])) ##-1 the last item

print(f"Average price per night: {sum(prices)/ len(prices)}")
```

```
Average price per night: 231.5
```

```
#modern python
import pandas as pd

df = pd.read_csv("hotel.csv")

df.head()
```

	hotel_id	name	city	star_rating	price_per_night	
0	101	Grand Plaza	New York	5	350	
1	102	Ocean View	Miami	4	220	
2	103	Mountain Retreat	Denver	4	185	
3	104	The Urban Stay	Chicago	3	140	
4	105	Sunset Resort	Los Angeles	5	410	

Next steps: [New interactive sheet](#)

```
df.price_per_night.mean()
```

```
np.float64(231.5)
```

```
df["price_per_night"].mean()
```

```
np.float64(231.5)
```

```
#write file + context manager
#native python
head = ["id","name","city"]
body = [
    [1, "CU","Bangkok"],
    [2, "LSE", "London"],
    [3, "Reading", "Reading"]

]

with open("school.csv","w") as file:
    writer = csv.writer(file)
    writer.writerow(head)
    writer.writerows(body) # body data more than 1 row
```

```
!ls
```

```
hotel.csv  sample_data  school.csv
```

```
!cat school.csv
```

```
id,name,city
1,CU,Bangkok
2,LSE,London
3,Reading,Reading
```

```
#modern python
import pandas as pd

df=pd.read_csv("school.csv")

df.head()
```

	id	name	city
0	1	CU	Bangkok
1	2	LSE	London
2	3	Reading	Reading

Next steps: [New interactive sheet](#)

```
df.to_csv("school2.csv")
```

```
!ls
```

```
hotel.csv  sample_data  school2.csv  school.csv
```

```
!cat school2.csv
```

```
,id,name,city
0,1,CU,Bangkok
1,2,LSE,London
2,3,Reading,Reading
```

```
##read data from internet
url = "https://raw.githubusercontent.com/mwaskom/seaborn-data/refs/heads/master/pengui
```

```
import pandas as pd
df_penguin = pd.read_csv(url)
```

```
df_penguin.tail()
```

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex
339	Gentoo	Biscoe	NaN	NaN	NaN	NaN	NaN
340	Gentoo	Biscoe	46.8	14.3	215.0	4850.0	FEMALE
341	Gentoo	Biscoe	50.4	15.7	222.0	5750.0	MALE
342	Gentoo	Biscoe	45.2	14.8	212.0	5200.0	FEMALE
343	Gentoo	Biscoe	49.9	16.1	213.0	5400.0	MALE



```
df_penguin.head()
```

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0	MALE
1	Adelie	Torgersen	39.5	17.4	186.0	3800.0	FEMALE
2	Adelie	Torgersen	40.3	18.0	195.0	3250.0	FEMALE
3	Adelie	Torgersen	NaN	NaN	NaN	NaN	NaN
4	Adelie	Torgersen	36.7	19.3	193.0	3450.0	FEMALE


Next steps: [New interactive sheet](#)

```
import pandas as pd
from pandas import read_csv
df_penguin = read_csv
```



```
## select column
df_penguin[["species", "island", "body_mass_g"]].head(10)
```

	species	island	body_mass_g	
0	Adelie	Torgersen	3750.0	
1	Adelie	Torgersen	3800.0	
2	Adelie	Torgersen	3250.0	
3	Adelie	Torgersen	NaN	
4	Adelie	Torgersen	3450.0	
5	Adelie	Torgersen	3650.0	
6	Adelie	Torgersen	3625.0	
7	Adelie	Torgersen	4675.0	
8	Adelie	Torgersen	3475.0	
9	Adelie	Torgersen	4250.0	

```
#filter rows
df_penguin[["species","island","body_mass_g"]].query("species == 'Adelie' and island =
```


	species	island	body_mass_g	
30	Adelie	Dream	3250.0	
31	Adelie	Dream	3900.0	
32	Adelie	Dream	3300.0	
33	Adelie	Dream	3900.0	
34	Adelie	Dream	3325.0	
35	Adelie	Dream	4150.0	
36	Adelie	Dream	3950.0	
37	Adelie	Dream	3550.0	
38	Adelie	Dream	3300.0	
39	Adelie	Dream	4650.0	
40	Adelie	Dream	3150.0	
41	Adelie	Dream	3900.0	
42	Adelie	Dream	3100.0	
43	Adelie	Dream	4400.0	
44	Adelie	Dream	3000.0	
45	Adelie	Dream	4600.0	
46	Adelie	Dream	3425.0	

```
##filter without query()  
df_penguin[df_penguin['body_mass_g']>=4500][["species","sex","body_mass_g"]]
```

	species	sex	body_mass_g	
7	Adelie	MALE	4675.0	
85	Adelie	Dream	3550.0	
17	Adelie	MALE	4500.0	
...	
39	Adelie	MALE	4650.0	
87	Adelie	Dream	3500.0	
45	Adelie	MALE	4600.0	
...	
81	Adelie	MALE	4700.0	
89	Adelie	Dream	3600.0	
...	
...	
338	Gentoo	FEMALE	4925.0	
91	Adelie	Dream	4300.0	
340	Gentoo	FEMALE	4850.0	
...	
341	Gentoo	MALE	5750.0	
93	Adelie	Dream	4450.0	
342	Gentoo	FEMALE	5200.0	
...	
343	Gentoo	MALE	5400.0	
95	Adelie	Dream	4300.0	
118 rows × 3 columns				
...	

Requests

97	Adelie	Dream	4350.0
98	Adelie	Dream	2900.0
99	Adelie	Dream	4100.0
132	Adelie	Dream	3500.0

How to get data via API the easy way

```
##request api

import requests

##api endpoint

url = "https://swapi.info/api/people/1"
```

```
response = requests.get(url)
##success response 200
```

```
140    Adelie   Dream      3400.0
```

```
response.json()["name"]
```

```
'142'    Adelie   Dream      3050.0
```

```
142    Adelie   Dream      3725.0
```

```
response.json()["height"]
```

```
'144'    Adelie   Dream      3650.0
```

```
##for loop
```

```
for i in range (10):
    print(i+1) ##python start with 0
```

```
145    Adelie   Dream      3700.0
```

```
1
2150    Adelie   Dream      3700.0
```

```
3
4151    Adelie   Dream      4000.0
```

```
5
6
7
8
9
10
```

```
##for loop
import requests
import time

characters =[]

for i in range(10):
    url= f"https://swapi.info/api/people/{i+1}"
    response = requests.get(url)
    response_json = response.json()
    data = [response_json["name"],
            response_json["height"],
            response_json["mass"]]
    characters.append(data)
    print(f"success: {i+1}")
    time.sleep(1)

print(characters)
```

```
success: 1
success: 2
success: 3
```

Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.