Libraries

- os
- random
- math
- functools (reduce)
- mailerpy
- requests, BeautifulSoup
- pandas
- matplotlib
- psycopg2 / pymysql (pip install psycopg2-binary)

OS

Built-in library that helps us to interact with OS

Common Methods:

os.path.join() ➔ Join one or more path component
os.rename() ➔ Renames file into new name (move)
os.remove() ➔ Deletes the file
os.path.getsize() ➔ Return size of file in bytes
os.mkdir() ➔ Makes folder # research on mkdirs
os.rmdir() ➔ Deletes the folder
os.path.exists() ➔ Checks if provided path exist
os.listdir() ➔ Shows number of files / folders present
os.system(cmd) ➔ Run system native command

random

Built-in library that helps us to perform random operation

Common Methods:

.choice(lis)	➔ Choose element from list/tuple
.sample(lis, n)	➔ Choose n element from list/tuple
.random()	➔ Generate float number between 0 & 1
.randint(a, b)	➔ Generate int number between a & b (inc)
.suffle(lis)	➔ Shuffles the list

math

Built-in library that helps us to perform math operation

Common Methods:

.floor(num)	➔ Gives floor of float
.ceil(num)	➔ Gives ceiling value of float
.pow(a,b)	➔ Generate a ^ b
.pi	➔ Returns value of PI
.radian(ang)	➔ Converts angle from degree to radian
.degrees(ang)	➔ Converts angle from radian to degree
.sin(ang)	➔ Sin value of ang (ang in radian). cos, sinh
.gcd(nums)	➔ Calculate GCD/HCF of numbers
.factorial(n)	➔ Calculates factorial of a number

Functools

- In-built library
- reduce is popular function from functools
- It is used with help of lambdas similar to map & filter

```
from functools import reduce

lis = [2, 4, 3, 5]
product = reduce(lambda x, y: x * y, lis)
print(product)
```

mailerpy

- External library. Installed as: `pip install mailerpy` in venv
- Used to send mail with ease
- Generate Password: https://myaccount.google.com/apppasswords

```
from mailerpy import Mailer
import os

PSPRT = os.environ.get("EMAIL_PASSWORD")
my_mailer = Mailer("smtp.gmail.com", "587", "email@gmail.com", PSPRT)

my_mailer.send_mail(to_list, subject, body, attachments=atch_list)
```

requests

- External library. Installed as: `pip install requests` in venv
- Used to interact with API

```
import requests

response = requests.get('https://www.google.com')
print(response.status_code)      # 200 ➔ OK
print(response.headers)
print(response.text)
```

BeautifulSoup

- External library. Installed as: `pip install beautifulsoup4` in venv
- Used for Web Scrapping for parsing HTML or XML doc

```
import requests
from bs4 import BeautifulSoup

response = requests.get('https://www.google.com')
soup = BeautifulSoup(response.text, "html.parser")
print(soup.title.text)
```

Scraping

```
import requests
from bs4 import BeautifulSoup

def get_data_from_table_soup(table):
        parsed_data = list()
        rows = table.find_all('tr')
        for row in rows:
                column = row.find_all(["td", "th"])
                data = [col.get_text(strip=True) for col in column]
                parsed_data.append(data)
        return parsed_data

response = requests.get("https://www.yetiairlines.com/flight-status")
if response.status_code != 200:
        print("Error on site")

soup = BeautifulSoup(resp.text, "html.parser")
table = soup.find("table", {"id": "flightInfo"})
flight_data = get_data_from_table_soup(table)
print(flight_data)      # Write to CSV File
```

datetime

- In-built library for managing date and time
- Common Functions

.datetime.now() ➔ Gives current date and time
.date.today() ➔ Give today's date [.year, .month, .day available]
dt_obj.strftime("%Y") ➔ Formats datetime object in given form
obj.strftime("%Y.%m") ➔ Formats datetime object in given form
.datetime.strptime() ➔ Converts string into datetime. Has string & format
dt_1 – dt_2 ➔ Find difference of two date & return delta

datetime

Some Codes		interpretation
%Y	→	Year
%m	→	Month as zero padded decimal
%-m	→	Month as decimal
%d	→	Day of month as zero padded decimal
%D	→	Day
%-d	→	Day of month as decimal
%B	→	Month Name
%b	→	Month abbreviated name
%y	→	Year in short form as zero padded decimal
%-y	→	Year in short form as decimal
%H	→	Hour 24 o'clock format
%I	→	Hour 12 o'clock format
%p	→	AM or PM
%M	→	Minute
%S	→	Second
%t	→	Microsecond
%z	→	UTC offset in form +HHMM or -HHMM
%Z	→	Timezone Name
%a	→	Abbreviated weekday name
%A	→	Full weekday name
%w	→	Weekday as decimal number
%j	→	Day of year

pandas

- External library. Installed as: `pip install pandas` in venv
- Used for data manipulation analyzing and cleaning

```
import pandas as pd

usr_df = pd.read_csv(file_name)
print(usr_df.head())
filtered_df = usr_df[usr_df.salary > 100000] # df[(condn_1) & (condn_2)]
filtered_df['company'] = 'ABC Inc'
filtered_df.to_csv(output_file, index=False)
print(user_df.student.sum())
print(user_df.groupby("course")["student"].sum()      # count, sum
```

matplotlib

- External library. Installed as: `pip install matplotlib` in venv
- Used for data visualization

```
import pandas as pd
import matplotlib.pyplot as plt

usr_df = pd.read_csv(file_name)
usr_df.plot(x='course', y='student', kind='bar')      # Group required
plt.title("Plot of Course and Student")
plt.xticks(rotation=0)
plt.show()
```

Exercise