23/12/2025

**Work done:**

- Modules in python
- Libraries

**Description:**

## Modules

- ➔ Module is file that contain python code such as : functions , variables , classes etc
- ➔ Module is used to organize large program
- ➔ Used for better readability
- ➔ Used for reuse code
- ➔ Type : 1) built in, 2)user defined

- ### Built in modules
    - ❖ **Math -** used for doing mathematical operations

      ```
      Import math

      print(math.sqrt(25)) # 5.0
      print(math.pow(2,3)) # 8
      print(math.pi) # 3.14….
      print(math.factorial(5)) # 120
      ```

    - ❖ **Random -** returns random number

      ```
      import random

      print(random.randint(1,10))
      print(random.choice[10,41,90])
      ```

    - ❖ **Datetime -** returns date , time , year , day

      ```
      import datetime

      today = datetime.date.today()
      print(today)
      ```

    - ❖ **os**

```
import os

print(os.getcwd()) # C:\Users\Sayma_kazi
```

- **User- defined modules**

**Save the code in mymodule.py file**

```
def greeting(name):
        print("hello " + name)
```

Using module

```
import mymodule
print(greeting("Sayma"))
```

**Variables in module**

module1.py

```
Person{
        "name" : "sayma"
        "age" : 18
        "Gender" : "female"
}
```

Using it

```
import module1

x = module1.person["name"]
print(x)
```

**Renaming the module**

module1.py

```
Person{
        "name" : "sayma"
        "age" : 18
        "Gender" : "female"
}

import module1 as m1
x = m1.person["name"]
```

```
print(x)
```

- ## Third party libraries

  - **NumPy -** Used for numerical operations and arrays.

    ```
    import numpy as np

    arr = np.array([1, 2, 3, 4])
    print(arr)  #[1 2 3 4]
    ```

  - **Pandas -** Used for data analysis.

    ```
    import pandas as pd

    data = {
        "Name": ["Asha", "Ravi"],
        "Marks": [85, 90]
    }

    df = pd.DataFrame(data)
    print(df)

    o/p:
       Name  Marks
    0  Asha     85
    1  Ravi     90
    ```

  - **Matplotlib -** Used for plotting graphs.

    ```
    import matplotlib.pyplot as plt

    x = [1, 2, 3]
    y = [2, 4, 6]

    plt.plot(x, y)
    plt.show()
    ```

- ## __name__  = “ __main__”

`__name__ == "__main__"` ensures that code runs only when the file is executed directly, not when imported.

file1.py

```
print(“This is from file1”)
if __name__  = “ __main__” ;
        print(“this file is running directly”)
```

If i execute this file directly the output will be:

```
This is from file1
this file is running directly
```

But if i run this code like this:

file2.py

```
import file1
print(“This is from file2”)
```

Then output will be

```
This is from file1
This is from file2
```

- ➔ `__name__` is a special variable
- ➔ `"__main__"` means **main program**
- ➔ Used to control execution
- ➔ Avoids unwanted code execution during import