Unit 3: Modules and Packages

Topic: Working with Modules and Packages in Python

1. Introduction

In Python, large programs are divided into small, manageable parts to make them easy to write, understand, and maintain. These parts are called modules and packages.

2. What is a Module?

Definition

A module is a file that contains Python code such as functions, variables, and classes, which can be reused in other programs.

Simple Definition:

A module is a Python file that stores related code so we can use it again in other programs.

Examples of Built-in Modules

- math – for mathematical operations
- random – for generating random numbers
- datetime – for working with date and time
- sys – for system-related functions

3. Creating a User-Defined Module

Steps

1. Create a Python file (example: mymodule.py)
2. Write functions or variables in it
3. Import it into another Python program

Example

File: mymodule.py

```
def greet(name):
return "Hello, " + name
```

Main Program

```
import mymodule
print(mymodule.greet("Student")
```

4. Ways to Import a Module

1. Import the Whole Module

```
import math
print(math.sqrt(25))
```

2. Import Specific Functions

```
from math import sqrt
print(sqrt(16))
```

3. Import with Alias

```
import math as m
print(m.pi)
```

5. What is a Package?

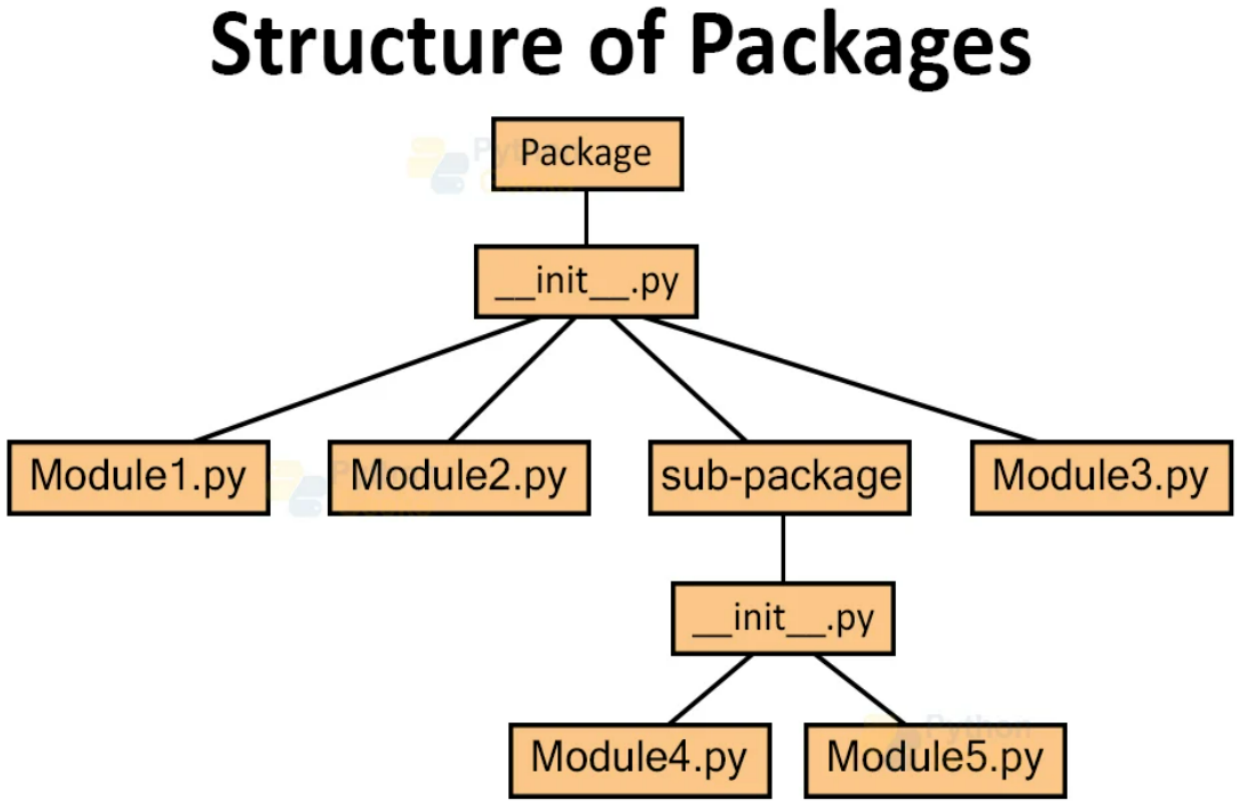
Definition

A package is a collection of related modules stored inside a folder (directory) to organize large Python programs.

Simple Definition

A package is a folder that contains multiple Python files (modules) to keep big programs well organized.

6. Structure of a Package



7. Using a Package

Import a Module from a Package

```
from my_package import module1
module1.function_name()
```

Import a Specific Function

```
from my_package.module1 import function_name
function_name()
```

8. Why Use Modules and Packages?

- Organizes large programs
- Improves code readability
- Avoids name conflicts
- Supports teamwork
- Makes debugging and maintenance easy
- Encourages code reusability

9. Difference Between Module and Package

Feature	Module	Package
Definition	A single Python file	A folder containing multiple modules
Extension	.py file	Directory (folder)
Size	Small program	Large program
Example	math.py	mypackage/

10. Built-in vs User-Defined

Built-in Modules

These are already available in Python.

Examples:

- math
- os
- sys
- random

User-Defined Modules

These are created by the programmer.

Examples:

- student.py
- calculator.py

Introduction to Popular Python Libraries for Specific Tasks

Topic: Python Libraries for Data Analysis, Web Development, and Game Development

1. What is a Python Library?

A **Python library** is a collection of pre-written code (modules and functions) that helps programmers perform common tasks easily and efficiently without writing everything from scratch.

Simple Definition (Student Friendly)

A Python library is a **toolbox** that contains ready-made programs to help us do specific work like analyzing data, building websites, or creating games.

Why Do We Use Libraries?

- Saves time
- Reduces coding effort
- Improves program quality
- Makes development faster
- Helps in professional projects

2. Categories of Popular Python Libraries

Python libraries are used in different fields. In this topic, we focus on three main areas:

1. Data Analysis
2. Web Development
3. Game Development

3. Python Libraries for Data Analysis

Data analysis means collecting, cleaning, and studying data to find useful information and patterns.

3.1 NumPy

Full Form: Numerical Python

Definition

NumPy is a Python library used for working with **numbers, arrays, and mathematical operations**.

Features

- Supports large arrays and matrices
- Performs fast mathematical calculations
- Used in scientific computing

Example Use

- Calculating averages
- Working with lists of numbers
- Performing matrix operations
-

Sample Code

```
import numpy as np
arr = np.array([10, 20, 30, 40])
print(arr)
```

3.2 Pandas

Definition

Pandas is a Python library used for handling and analyzing structured data such as tables and files.

Features

- Works with Excel and CSV files
- Supports data cleaning
- Allows filtering and sorting data

Example Use

- Student marks analysis
- Attendance records
- Sales data processing

Sample Code

```
import pandas as pd
data = {
'Name': ['Amit', 'Riya', 'Suman'],
'Marks': [85, 90, 78]
}
df = pd.DataFrame(data)
print(df)
```

3.3 Matplotlib

Definition

Matplotlib is a Python library used to draw graphs and charts.

Features

- Creates line charts, bar charts, and pie charts
- Helps in data visualization
- Makes reports more understandable

Example Use

- Showing exam performance in graphs
- Visualizing sales growth

Sample Code

```
import matplotlib.pyplot as plt
x = [1, 2, 3, 4]
y = [10, 20, 25, 30]
plt.plot(x, y)
plt.show()
```

4. Python Libraries for Web Development

Web development means **creating websites and web applications using Python**.

4.1 Flask

Definition

Flask is a lightweight Python library used to **build simple and small web applications**.

Features

- Easy to learn
- Flexible
- Suitable for beginners

Example Use

```
from flask import Flask
app = Flask(__name__)
@app.route("/")
def home():
    return "Welcome to My Website"
app.run()
```

4.2 Django

Definition

Django is a high-level Python library used to **develop large and secure web applications**.

Features

- Built-in security
- Database support
- User authentication system

Example Use

- Online shopping websites
- College management systems

5. Python Libraries for Game Development

Game development means **creating computer games using programming**.

5.1 Pygame

Definition

Pygame is a Python library used to **create 2D games and multimedia applications**.

Features

- Supports graphics and sound
- Keyboard and mouse input
- Easy for beginners

Example Use

- Snake game
- Car racing game

Sample Code

```
import pygame
pygame.init()
screen = pygame.display.set_mode((400, 300))
pygame.display.set_caption("My First Game")
running = True
while running:
for event in pygame.event.get():
if event.type == pygame.QUIT:
running = False
pygame.quit()
```

PyCharm IDE, Git Integration, PyTest, and Database Connectivity

Topic: Working with PyCharm IDE, Git, Testing, and Databases in Python

1. Introduction

Modern Python development is not only about writing code but also about **using professional tools** to manage, test, and connect programs to databases. In this topic, students will learn about:

- PyCharm IDE
- Git and Git Integration
- PyTest for testing
- Python connectivity with MySQL and MongoDB
- CRUD Operations

2. What is an IDE?

Definition

An **IDE (Integrated Development Environment)** is a software application that provides tools to write, run, test, and debug programs in one place.

Simple Definition

An IDE is a **software tool where we write and run our Python programs easily**.

Features of an IDE

- Code editor
- Run and debug tools
- Error highlighting
- File and project management

3. PyCharm IDE

Definition

PyCharm is a popular Python IDE developed by **JetBrains** that helps programmers write, test, and debug Python programs efficiently.

Features of PyCharm

- Smart code completion
- Error detection
- Debugging tools
- Project structure management
- Git integration

Uses

- Developing Python applications
- Web development using Django and Flask
- Data science and testing projects

4. Introduction to Git

Definition

Git is a **version control system** used to track changes in code and manage teamwork in software development.

Simple Definition

Git helps programmers **save different versions of their programs and work together safely**.

Uses of Git

- Tracks code changes
- Helps in teamwork
- Allows rollback to old versions
- Stores projects online using GitHub

5. Git Integration with PyCharm

Meaning

Git integration means **using Git features directly inside PyCharm** without using the command line.

Common Git Operations in PyCharm

- Initialize Git repository
- Commit changes
- Push to GitHub
- Pull updates
- View file history

Steps (Basic)

1. Open project in PyCharm
2. Go to **VCS → Enable Version Control Integration**
3. Select Git
4. Connect to GitHub
5. Commit and push changes

6. Introduction to PyTest

Definition

PyTest is a Python testing framework used to **check whether programs work correctly**.

Simple Definition

PyTest is a tool that helps us **test our Python code automatically**.

Features

- Simple syntax
- Automatic test discovery
- Detailed test reports

Example:

```
def add(a, b):
    return a + b
def test_add():
    assert add(2, 3) == 5
```

7. Python Connectivity with Databases

Meaning

Python can connect to databases to **store, retrieve, update, and delete data**.

8. MySQL Database Connectivity

Definition

MySQL is a **relational database** used to store data in tables.

Python Library Used

- mysql-connector-python

Example Code

```
import mysql.connector
conn = mysql.connector.connect(
host="localhost",
user="root",
password="password",
database="college"
)
cursor = conn.cursor()
cursor.execute("SELECT * FROM students")
for row in cursor:
print(row)
conn.close()
```

9. MongoDB Database Connectivity

Definition

MongoDB is a **NoSQL database** that stores data in the form of documents instead of tables.

Python Library Used

- pymongo

Example Code

```
from pymongo import MongoClient
client = MongoClient("mongodb://localhost:27017/")
db = client["college"]
collection = db["students"]
for student in collection.find():
    print(student)
```

10. CRUD Operations

Meaning

CRUD stands for:

- **C** – Create (Insert Data)
- **R** – Read (View Data)
- **U** – Update (Modify Data)
- **D** – Delete (Remove Data)

11. CRUD Example (MySQL – Simple)

Create (Insert)

```
cursor.execute("INSERT INTO students VALUES (1, 'Amit')")
conn.commit()
```

Read (Select)

```
cursor.execute("SELECT * FROM students")
```

Update

```
cursor.execute("UPDATE students SET name='Riya' WHERE id=1")
conn.commit()
```

Delete

```
cursor.execute("DELETE FROM students WHERE id=1")
conn.commit()
```

*****