🐍 PYTHON SYLLABUS (BEGINNER → ADVANCED)

📘 Module 1: Introduction & Setup

1.1 Python Basics

What is Python?
History
Features
Applications (Web, Data Science, AI, Automation)

1.2 Installation & Environment

Installing Python
Python IDLE
PyCharm
VS Code
Running Python programs

1.3 First Python Program

`print()` statement
Comments (single-line, multi-line)
Python syntax basics

1.4 Variables & Data Types

Variables and memory concept
Data Types:
Integer
Float

String
Boolean

1.5 Keywords & Identifiers

Python keywords
Identifiers
Naming conventions
Best practices

1.6 User Input & Type Casting

`input()` function
Type conversion:
`int()`
`float()`
`str()`

📓 Module 2: Operators & Control Flow

2.1 Operators

Arithmetic Operators
Assignment Operators
Comparison Operators
Logical Operators
Membership Operators
Identity Operators
Bitwise Operators

2.2 Conditional Statements

`if` statement
`if-else`
`elif`
Nested `if-else`

2.3 Loops

For Loop

- `for` loop syntax
- `range()` function
- Iteration logic

While Loop

- `while` loop
- Infinite loop concept

Loop Control Statements

- `break`
- `continue`
- `pass`

📓 Module 3: String Manipulation

3.1 String Basics

- String creation
- String immutability

3.2 Indexing & Slicing

- Positive indexing
- Negative indexing
- String slicing

3.3 String Iteration

- Looping through strings

3.4 String Functions

- `lower()`, `upper()`
- `find()`
- `replace()`
- `split()`
- `strip()`
- `format()`

3.5 String Operations

- String concatenation
- String formatting (f-strings)

Module 4: Python Data Structures (Core)

4.1 Lists

- Creating lists
- Indexing & slicing
- List comprehension
- List methods:
  - `append()`
  - `extend()`
  - `insert()`
  - `pop()`
  - `remove()`

4.2 Tuples

- Tuple creation
- Difference between List & Tuple
- Tuple packing & unpacking

4.3 Dictionaries

- Key–Value pairs

Dictionary methods:
`get()`
`keys()`
`items()`
`update()`
Nested dictionaries

4.4 Sets

Creating sets
Set operations:
Union
Intersection
Difference
Set methods:
`add()`
`discard()`

4.5 Stack & Queue

Stack implementation using List
Queue implementation using List
Real-world examples

📓 Module 5: Functions, Recursion & Modules

5.1 User Defined Functions

Function definition
Function call
Parameters & Arguments
Default arguments
Keyword arguments
Variable-length arguments (`*args`, `**kwargs`)
Return statement

5.2 Scope of Variables

Local variables
Global variables

🔁 5.3 Recursion (ADDED)

What is recursion?
Base condition
Recursive case
Recursive call stack
Difference between recursion & iteration
Examples:
Factorial
Fibonacci
Sum of digits
Advantages & limitations of recursion

5.4 Modules

What is a module?
Importing built-in modules:
`math`
`random`
`datetime`
Creating custom modules
`__name__ == "__main__"` concept

📱 Module 6: Object-Oriented Programming (OOPs)

6.1 OOP Fundamentals

Class & Object

Constructors (`__init__` method)
Instance variables

6.2 Inheritance

Single inheritance
Multi-level inheritance
Multiple inheritance

6.3 Encapsulation

Private variables
Getters & Setters

6.4 Polymorphism

Method overloading (concept)
Method overriding

📓 Module 7: Advanced Topics

7.1 Exception Handling

Errors vs Exceptions
`try` block
`except` block
`else` block
`finally` block

7.2 File Handling

File modes:
Read (`r`)
Write (`w`)
Append (`a`)
Text vs Binary files

File operations

7.3 Pickle Module

Serialization
Deserialization
Use cases

7.4 JSON Handling

Reading JSON data
Writing JSON data
Real-world usage

7.5 Regular Expressions (RegEx)

Pattern matching basics
Common regex functions
Validation use cases

☑ Outcome of This Syllabus

After completing this playlist, learners will:

Master Python fundamentals
Write structured & reusable code
Understand recursion deeply
Apply OOP concepts
Handle files & errors confidently
Be ready for **interviews, projects, and next-level domains**