Python Course Syllabus

1: Python Introduction (2 hours)

- What is a programming language? Overview of high-level vs. low-level languages.
- Introduction to Python: History, features, and applications (web, data science, automation).
- Python as an interpreted language: How Python executes code.
- Setting up Python: Installing Python and an IDE (e.g., VS Code, PyCharm).
- Virtual environments: Purpose and basic setup using `venv`.

2: Data Types, Variables and Strings (2 hours)

- Python data types: `int`, `float`, `str`, `bool`, `None`.
- Variables: Declaration, naming conventions, and dynamic typing.
- Operator precedence: Arithmetic, comparison, logical operators.
- Strings: Creation, concatenation, and methods.
- Escape sequences and formatted strings (f-strings, `.format()`).
- String indexing, slicing, and immutability.
- Type conversion: `int()`, `str()`, `float()`.

3: Data Structures (2 hours)

- Lists: Creation, indexing, slicing, and methods (e.g., `append`, `remove`).
- Dictionaries: Key-value pairs, accessing, and modifying.
- Sets: Unique elements, set operations (union, intersection).
- Tuples: Immutable sequences, use cases.

4: Conditionals and Loops (2 hours)

- Conditionals: `if`, `elif`, `else`, and indentation.
- Logical operators: `and`, `or`, `not`.
- `is` vs. `==`: Identity vs. equality.
- Introduction to loops: `for` and `while`.
- Loop control: `break`, `continue`, `pass`.
- `range()` and `enumerate()` for iteration.

5: Functions (2 hours)

- Functions: Definition, parameters, return statements.
- Default parameters, keyword arguments, and docstrings.
- `*args` and `**kwargs`: Variable-length arguments.
- Scope: Local, global, and nonlocal variables.
- Methods vs. functions: Object-oriented vs. standalone.

Minor Project 1:

- Develop a simple calculator program using functions to perform basic operations (add, subtract, multiply, divide) with user input.

6: Object-Oriented Programming (4 hours)

- What is OOP? Classes and objects.
- `__init__`: Constructor method.
- Four pillars of OOP: Encapsulation, abstraction, inheritance, polymorphism.
- `super()`: Accessing parent class methods.
- Multiple inheritance: Combining multiple parent classes.
- Dunder methods: `__str__`, `__repr__`, `__add__`, etc.

Minor Project 2:

- Build a bank account management system using classes to handle deposits, withdrawals, and balance inquiries.

7: Useful Functions (2 hours)

- Pure functions: Characteristics and benefits.
- `map()`, `filter()`, `zip()`, and `reduce()`: Functional programming tools.
- Lambda expressions: Anonymous functions.

8: Decorators and Generators (2 hours)

- Decorators: Enhancing function behavior.
- Generators: `yield` keyword and lazy evaluation.

9: Error Handling, Logging and Debugging (2 hours)

- Error handling: `try`, `except`, `else`, `finally`.

- Logging: Setting up basic logging for debugging.
- Debugging: Using print statements and IDE debuggers.

10: Modules, File I/O (2 hours)

- File I/O: Reading/writing `.txt`, `.json`, `.csv` files.
- Modules: Importing custom, standard, and third-party modules.

Minor Project 3:

- Develop a program to read student grades from a `.csv` file, calculate averages, and log results to a file. Perform Error Handling (FILE NOT FOUND Error)

11: Programming session (2 hours)

- Interview Coding Exams
- Interview MCQs

12: Introduction to Data Science (4 hours):

- Use numpy for array operations, pandas for data manipulation, and matplotlib for plotting.

Minor Project 4:

- Data Analysis and Visualization: House Price Dataset

13: Introduction to Machine Learning (2 hours):

- Introduction to machine learning.
- Overview of scikit-learn and a simple linear regression model.
- Overview of Streamlit

Minor Project 5:

- Build a basic ML model to predict house prices (dataset provided).
- Develop a Streamlit app for house price prediction.