**Introduction to Object-Oriented Programming**

**Basic concepts:** classes, objects, methods, and attributes

Differences between procedural and object-oriented programming

**Classes and Objects**

Defining classes

Creating instances (objects) from classes

The \_\_init\_\_ method

**Attributes and Methods**

Instance attributes

Class attributes

Instance methods

Class methods

Static methods

The self parameter

**Encapsulation**

Public, protected, and private attributes

Getter and setter methods

Property decorators (@property)

**Inheritance**

Single inheritance

Multiple inheritance

Method overriding

The super() function

**Polymorphism**

Method overloading

Method overriding

Duck typing

**Abstraction**

Abstract base classes (ABCs)

The abc module

**Magic Methods and Operator Overloading**

Common magic methods (\_\_str\_\_, \_\_repr\_\_, \_\_eq\_\_, \_\_lt\_\_, etc.)

Customizing behavior with magic methods

**Composition vs. Inheritance**

When to use composition over inheritance

Building complex objects using composition

**Metaprogramming**

Metaclasses

The type() function

Dynamic class creation

**Testing Object-Oriented Code**

Writing unit tests for classes and methods

Using unittest or pytest frameworks

**Decorators**

Function decorators

Class decorators

**Serialization and Deserialization**

Using pickle, json, and other libraries to serialize objects

**Exceptions and Error Handling**

Custom exceptions

Using try, except, finally blocks in OOP

**The \_\_main\_\_ block**

Writing executable Python scripts with object-oriented code

**Practical OOP Projects in Python**

Real-world examples and projects using OOP principles