Lesson **12 : OOP**

**Research Work**

1. **How does object-oriented programming differ from procedural programming?**

Object-oriented programming organizes code using objects that combine data and functions, making it easier to manage, reuse, and scale. Procedural programming, on the other hand, follows a step-by-step approach using functions and procedures. OOP is better suited for complex applications, while procedural programming works well for simpler tasks.

1. **What is a constructor in Python? What happens if it is not defined explicitly?**

In Python, a **constructor** is a special method called \_\_init\_\_() that is automatically invoked when a new object of a class is created. It is used to initialize the object's attributes with specific values. If a constructor is **not defined explicitly**, Python provides a **default constructor** that does nothing but creates the object without initializing any attributes. This means you can still create objects, but you'll need to set their attributes manually after creation.

1. **Explain the concept of class vs instance variables with example.**

In Python, class variables are shared by all instances of a class, meaning they have the same value for every object. In contrast, instance variables are unique to each object and can hold different values. Class variables are defined outside any method, while instance variables are usually set inside the \_\_init\_\_() constructor using self. This distinction helps manage shared data versus object-specific data.

1. **What is the purpose of \_str() and how is it different from \_\_repr\_()?**

The \_\_str\_\_() method in Python is used to define a human-readable string representation of an object, which is what gets returned when you use print() or str() on the object. In contrast, the \_\_repr\_\_() method is meant to provide an unambiguous string representation of the object, often used for debugging and development, and ideally should be a valid Python expression that can recreate the object. If \_\_str\_\_() is not defined, Python will use \_\_repr\_\_() as a fallback.

**5. Why is it beneficial to use object-oriented programming in larger projects?**

Object-oriented programming (OOP) is beneficial in larger projects because it helps organize complex code by breaking it into reusable and modular components called objects. It promotes better code structure through concepts like inheritance, encapsulation, and polymorphism, making it easier to manage, scale, and maintain. OOP also improves collaboration in teams, as different developers can work on different classes or modules without interfering with each other’s code.