Q.What is OOPS concept where we have to use oops concept in our automation framework?

Ans: OOPs concepts, such as **encapsulation, inheritance, polymorphism, and abstraction,** play a crucial role in designing and implementing an Automation Framework. These concepts allow for **modular code organisation, code reuse, flexibility, and maintainability**. Each OOPs concept is applied to enhance the effectiveness and efficiency of an Automation Framework. By applying OOPs concepts effectively, developers can achieve modularity, code reusability, flexibility, and scalability in their test automation solutions.

**1. ABSTRACTION**

Abstraction is a methodology of hiding the implementation and displaying the functionality to end users.

Example: In **Page Object Model** design pattern**,** we write locators (such as Id, Name, Xpath, etc. ) and the methods in a Page class. Then we utilise these locators and methods in tests but we can’t see the implementation of the methods

**2. Interface**

We all are aware of this statement

**WebDriver driver = new FirefoxDriver();**

Here **WebDriver** is an Interface. In the above statement, we are initialising the Firefox browser using Selenium WebDriver (Interface).

It means we are creating a reference variable (driver) of the interface (WebDriver) and creating an object of FirefoxDriver class.

### 3. INHERITANCE

Mechanism by which one class acquires the properties (instance variables) and functionalities of another class is known as **Inheritance**.

Base Class in the Automation Framework is used to initialise the WebDriver interface, WebDriver waits, Property files, Excels, etc. in the Base class.

We extend the Base class to other classes such as Tests and Utility class.

### 3. POLYMORPHISM

Polymorphism allows us to perform a task in multiple ways.

A combination of overloading and overriding is known as Polymorphism.

Let's understand overloading and overriding in details.

a)Method Overloading:We use Implicit wait in Selenium. The implicit wait is an example of overloading.

In **Implicit wait**, we use different time stamps such as SECONDS, MINUTES, HOURS, etc.,

**Action** class in TestNG is also an example of overloading.

**Assert** class in TestNG is also an example of overloading.

b). METHOD OVERRIDING

We use a method that was already implemented in another class by changing its parameters. To understand this you need to understand Overriding in Java.

Declaring a method in a child class that is already present in the parent class is called Method Overriding.

### 5. ENCAPSULATION

All the classes in a framework are an example of Encapsulation.

In POM classes, we declare the data members using @FindBy, and initialization of data members will be done using Constructor to utilise those in methods.We use Getter setter method

Encapsulation is a mechanism of binding code and data (variables) together in a single unit.