# **MODULE – 4(Advance PHP)**

1. **• What Is Object Oriented Programming?**

**Ans-**OOP stands for Object-Oriented Programming.

Procedural programming is about writing procedures or functions that perform operations on the data, while object-oriented programming is about creating objects that contain both data and functions.

Object-oriented programming has several advantages over procedural programming:

* OOP is faster and easier to execute
* OOP provides a clear structure for the programs
* OOP helps to keep the PHP code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug
* OOP makes it possible to create full reusable applications with less code and shorter development time

**2 What Are Properties Of Object Oriented Systems?**

**Ans-**

**Encapsulation:**

* **Definition:** Encapsulation refers to the bundling of data (attributes or properties) and the methods (functions or procedures) that operate on that data into a single unit known as a class.

**Abstraction:**

* **Definition:** Abstraction involves simplifying complex systems by modeling classes based on the essential properties and behaviors an object should have.

**Inheritance:**

* **Definition:** Inheritance is a mechanism that allows a class (subclass or derived class) to inherit the properties and behaviors of another class (superclass or base class).

**Polymorphism:**

* **Definition:** Polymorphism allows objects of different classes to be treated as objects of a common base class. It enables a single interface to represent different types.

**Polymorphism:**

* **Definition:** Polymorphism allows objects of different classes to be treated as objects of a common base class. It enables a single interface to represent different types.

**Message Passing:**

* **Definition:** Objects in an object-oriented system communicate with each other by sending messages. A message typically involves invoking a method or requesting some action.

**3• What Is Difference Between Class And Interface?**

* **Ans-Class:** A class is a blueprint or template for creating objects. It defines properties (attributes) and methods (functions) that the objects of the class will have.
* **Interface:** An interface, on the other hand, defines a contract for classes that implement it. It specifies a set of method signatures that the implementing classes must provide.

**4. • What Is Overloading?**

**Ans-**Overloading in PHP provides means to dynamically create properties and methods. These dynamic entities are processed via magic methods one can establish in a class for various action types.

The overloading methods are invoked when interacting with properties or methods that have not been declared or are not [visible](https://www.php.net/manual/en/language.oop5.visibility.php) in the current scope. The rest of this section will use the terms inaccessible properties and inaccessible methods to refer to this combination of declaration and visibility.

All overloading methods must be defined as public.

**5. • What Is T\_PAAMAYIM\_NEKUDOTAYIM (Scope Resolution Operator (::) with Example**

**Ans-**This operator is known as Paamayim Nekudotayim named in Hebrew means double colon. In PHP, the errors that occurred related to this scope resolution operator will be displayed in the browser using this name only, that is T\_PAAMAYIM\_NEKUDOTAYIM, a PHP error constant that denotes improper code lines in this regard

**6. • What are the differences between abstract classes and interfaces?**

**Ans-**Interface are similar to abstract classes. The difference between interfaces and abstract classes are: Interfaces cannot have properties, while abstract classes can. All interface methods must be public, while abstract class methods is public or protected.

**7. • Define Constructor and Destructor?**

Constructor and destructor are special methods in object-oriented programming (OOP) that are invoked when an object is created or destroyed. In PHP, you can use constructor and destructor to manage the initialization and cleanup of your objects, as well as implement dependency injection and type hinting

**8 • How to Load Classes in PHP?**

**Ans-**In PHP, classes are typically organized in files, and these files need to be included or autoloaded to use the classes in your code. There are a few ways to load classes in PHP:

1. **Include or Require Statements:**
   * Use the **include** or **require** statements to include the file containing the class definition.

**Autoloading:**

* PHP supports autoloading, where classes are loaded automatically when they are needed, eliminating the need for manual includes.
* Use the **spl\_autoload\_register** function to register an autoloader function.

In PHP, classes are typically organized in files, and these files need to be included or autoloaded to use the classes in your code. There are a few ways to load classes in PHP:

1. **Include or Require Statements:**
   * Use the **include** or **require** statements to include the file containing the class definition.
   * Example:

phpCopy code

// Include file with class definition include('path/to/YourClass.php'); // Create an instance of the class $obj = new YourClass();

1. **Autoloading:**
   * PHP supports autoloading, where classes are loaded automatically when they are needed, eliminating the need for manual includes.
   * Use the **spl\_autoload\_register** function to register an autoloader function.
   * Example:

phpCopy code

// Autoloader function function myAutoloader($class) { include 'path/to/' . $class . '.php'; } // Register the autoloader function spl\_autoload\_register('myAutoloader'); // Create an instance of the class (autoloads if not already included) $obj = new YourClass();

1. **Composer Autoloading:**
   * If you are using Composer (a dependency manager for PHP), it provides a convenient autoloading mechanism.
   * Define your autoloading information in the **composer.json** file, and Composer will generate an autoloader for you.

**Namespace and Use Statements:**

* If you are using namespaces, use the **namespace** and **use** statements to organize and reference your classes.

9. • How to Call Parent Constructor?

Ans-in order to run a parent constructor, a call to parent::\_\_construct() within the child constructor is required. If the child does not define a constructor then it may be inherited from the parent class just like a normal class method (if it was not declared as private).

**10Are Parent Constructor Called Implicitly When Create An ObjectOf Class?**

**Ans-**In PHP, when you create an object of a child class, the parent class constructor is not called implicitly by default. However, you can explicitly call the parent class constructor using the **parent::\_\_construct()** syntax within the child class constructor if you want to execute the parent class constructor as part of the child class instantiation process.

**11• What Happen, If Constructor Is Defined As Private Or Protected?**

**Ans-**if a constructor is defined as private or protected in a class, it affects the accessibility of the constructor and, consequently, how instances of that class can be created.

Here are the implications of defining a constructor as private or protected:

1. **Private Constructor:**
   * If a constructor is declared as private, it means that the constructor is accessible only within the class where it is defined. It cannot be called from outside the class or by its child classes.

**12. • What are PHP Magic Methods/Functions? List them Write program for Static Keyword in PHP?**

Methods that begin with 2 underscores(\_\_) are generally called Magic methods in PHP. These methods names are limited to some list of [PHP](https://www.php.net/manual/en/intro-whatis.php) supported keywords that are reserved. So any function should not be defined with the name of PHP magic methods.

Usually, these functions should be defined by the user and there is no need to call them explicitly.

* \_\_construct()
* \_\_destruct()
* \_\_call($fun, $arg)
* \_\_callStatic($fun, $arg)
* \_\_get($property)
* \_\_set($property, $value)
* \_\_isset($content)
* \_\_unset($content)

**13. • Create multiple Traits and use it in to a single class?**

**Ans-**PHP implements a way to reuse code called Traits.

Traits are a mechanism for code reuse in single inheritance languages such as PHP. A Trait is intended to reduce some limitations of single inheritance by enabling a developer to reuse sets of methods freely in several independent classes living in different class hierarchies. The semantics of the combination of Traits and classes is defined in a way which reduces complexity, and avoids the typical problems associated with multiple inheritance and Mixins.

A Trait is similar to a class, but only intended to group functionality in a fine-grained and consistent way. It is not possible to instantiate a Trait on its own. It is an addition to traditional inheritance and enables horizontal composition of behavior; that is, the application of class members without requiring inheritance.

**14. • Use of The $this keyword**

Ans-The **$this** keyword in PHP is a special variable that refers to the current object instance within a class. It is used to access properties and methods of the current object. Here are some common use cases of the **$this** keyword in PHP:

15