Object-Oriented Programming.

In PHP.

What is OOP?

OOP stands for Object-Oriented Programming, which is a programming paradigm that uses objects – instances of classes – to organize and structure code.

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.

Class: A class is user define datatype. A class is a blueprint or template for creating objects. It defines a set of attributes and methods that the objects of the class will have.

Object: An object is an instance of a class. It is a concrete entity created from the blueprint defined by a class.

4 pillars of OOP

1. **Encapsulation:** Encapsulation is one of the fundamental principles of Object-Oriented Programming (OOP). It involves bundling the data (attributes) and the methods (functions) that operate on the data into a single unit known as a class. The class serves as a blueprint for creating objects, and encapsulation helps in hiding the internal implementation details of the class from the outside world.

2. **Inheritance:** it is possible to inherit attributes and methods from one class to another.

Type of inheritance:

Single Inheritance: The inheritance in which a single derived class is inherited from a single base class

Multilevel Inheritance: A class can also be derived from one class, which is already derived from another class.

Multiple Inheritance: A class can also be derived from more than one base class, using a **comma-separated list**

Hierarchical Inheritance: Multiple classes inherit from a single superclass.

3. **Polymorphism:** The word "polymorphism" means having many forms. In simple words, we can define polymorphism as the ability of a message to be displayed in more than one form. A real-life example of polymorphism is a person who at the same time can have different characteristics. A man at the same time is a father, a husband, and an employee. So the same person exhibits different behavior in different situations. This is called polymorphism. Polymorphism is considered one of the important features of Object-Oriented Programming.

Types of Polymorphism

1. Compile-time Polymorphism

2. Runtime Polymorphism

4. **Abstraction:**

Abstraction is one of the key principles of Object-Oriented Programming (OOP) that allows you to model the real world in a simplified way. It involves hiding the complex implementation details and exposing only the essential features of an object. In other words, abstraction allows you to focus on what an object does rather than how it does it.

Constructor:

In PHP, the **__construct** function is a special method that is automatically called when an object is created from a class. It is a constructor method, and its primary purpose is to initialize the properties (attributes) of the object or perform any setup operations needed for the object to be in a valid state.

Destructor:

In PHP, a destructor is a special method called **__destruct** that is automatically invoked when an object is destroyed or goes out of scope. The primary purpose of a destructor is to perform cleanup tasks, such as releasing resources or closing connections, before the object is removed from memory

PHP - Access Modifiers: Properties and methods can have access modifiers which control where they can be accessed.

There are three access modifiers:

- Public - the property or method can be accessed from everywhere. This is default.
- protected - the property or method can be accessed within the class and by classes derived from that class
- Private - the property or method can ONLY be accessed within the class.

Class Constants:

Class constants can be useful if you need to define some constant data within a class.

A class constant is declared inside a class with the const keyword.

A constant cannot be changed once it is declared.

Interface: An interface is a way to define a contract for classes. It defines a set of methods that a class implementing the interface must implement. Interfaces provide a mechanism for achieving abstraction, and they help in enforcing a common structure for classes that need to adhere to a specific set of behaviors.

Traits: PHP only supports single inheritance: a child class can inherit only from one single parent.

So, what if a class needs to inherit multiple behaviors? OOP traits solve this problem.

Traits are used to declare methods that can be used in multiple classes. Traits can have methods and abstract methods that can be used in multiple classes, and the methods can have any access modifier (public, private, or protected).

Traits are declared with the `trait` keyword:

Static Methods:

Static methods can be called directly - without creating an instance of the class first.

Static methods are declared with the `static` keyword:

Static Properties:

Static properties can be called directly - without creating an instance of a class.

Static properties are declared with the `static` keyword: