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

Object-oriented programming (OOP) is a programming paradigm based on the concept of "objects", which can contain data, in the form of fields or attributes, and code, in the form of procedures or methods.

1. **What Are Properties Of Object Oriented Systems?**

Object-oriented systems possess several key properties that distinguish them from other programming paradigms. These properties include:

* Encapsulation
* Abstraction
* Inheritance
* Polymorphism
* Modularity
* Message Passing

1. **What Is Difference Between Class And Interface?**

In object-oriented programming, both classes and interfaces are fundamental concepts, but they serve different purposes and have distinct characteristics:

* **Class:**
* A class is a blueprint for creating objects in a programming language. It defines the properties (attributes) and behaviors (methods) that objects of that class will have.
* Objects are instances of classes, meaning they are specific instances created based on the structure defined by the class.
* Classes can contain member variables (fields) to hold data and member functions (methods) to perform operations on that data.
* Classes can also have constructors, which are special methods used for initializing objects when they are created.
* Classes support inheritance, allowing one class to inherit properties and behaviors from another class (base class or superclass).
* **Interface:**
* An interface is a reference type in many programming languages, similar to a class that can contain only constants, method signatures, default methods, static methods, and nested types.
* Interfaces define a contract for classes that implement them, specifying what methods a class must implement but not providing any implementation details.
* Classes can implement one or more interfaces, meaning they must provide implementations for all methods defined by those interfaces.
* Interfaces are used to achieve abstraction and provide a way to define a set of methods that multiple classes can adhere to, allowing for polymorphism.
* Unlike classes, interfaces cannot be instantiated directly; they serve as a blueprint for implementing classes.

1. **What Is Overloading?**

In object-oriented programming (OOP), overloading is a powerful feature that allows a class to have multiple methods with the same name but different parameters. It enables developers to write more concise and flexible code, enhancing the overall readability and maintainability of the program.

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

In PHP, T\_PAAMAYIM\_NEKUDOTAYIM is the name for the scope resolution operator, which means "double colon" in Hebrew. The scope resolution operator is a token that gives access to static, constant, and overridden properties or methods of a class. When referencing these items from outside the class definition, use the name of the class.

* Call for demo's version of func(): demo::func()
* Call for child's version of func(): Child::func()

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

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Abstract Class** | **Interface** |
| 1. | An abstract class can contain both abstract and non-abstract methods. | Interface contains only abstract methods. |
| 2. | An abstract class can have all four; static, non-static and final, non-final variables. | Only final and static variables are used. |
| 3. | To declare abstract class abstract keywords are used. | The interface can be declared with the interface keyword. |
| 4. | It supports multiple inheritance. | It does not support multiple inheritance. |
| 5. | The keyword ‘extend’ is used to extend an abstract class | The keyword implement is used to implement the interface. |
| 6. | It has class members like private and protected, etc. | It has class member’s public by default. |

1. **Define Constructor and Destructor?**

Constructors and destructors are special member functions of classes that are used to construct and destroy class objects. Construction may involve memory allocation and initialization for objects. Destruction may involve clean-up and deal location of memory for objects.

1. **How to Load Classes in PHP?**

PHP parser loads it automatically, if it is registered with spl\_autoload\_register() function. PHP parser gets the least chance to load class/interface before emitting an error. Syntax: spl\_autoload\_register(function ($class\_name) { include $class\_name .

1. **How to Call Parent Constructor?**

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). $obj = new OtherSubClass();

1. **Are Parent Constructor Called Implicitly When Create An Object Of Class?**

Parent constructors are not called implicitly if the child class defines a constructor. In order to run a parent constructor, a call to parent::\_\_construct() within the child constructor is required.

1. **What Happen, If Constructor Is Defined As Private Or Protected?**

The constructor may be made private or protected to prevent it from being called externally. If so, only a static method will be able to instantiate the class. Because they are in the same class definition they have access to private methods, even if not of the same object instance.

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

PHP magic methods are special methods that are automatically called when certain conditions are met. They can be used to intercept and handle property access, assignment, string conversion, and object invocation. This allows developers to customize the behavior of objects in a flexible and expressive manner.

**Here is a list of commonly used magic methods in PHP:**

* \_\_construct(): This magic method is the constructor of a class and is automatically called when an object is created. It is used to initialize object properties and perform any necessary setup.
* \_\_destruct(): The \_\_destruct() magic method is called automatically when an object is no longer referenced or when the script finishes execution. It is used to perform cleanup tasks or release resources held by the object.
* \_\_get(): This magic method is invoked when accessing inaccessible or non-existent properties of an object.
* \_\_set(): This magic method is invoked when assigning a value to an inaccessible or non-existent property of an object.
* \_\_isset(): This magic method is invoked when checking whether a property exists on an object.
* \_\_unset(): This magic method is invoked when unsetting a property on an object.
* \_\_call(): This magic method is invoked when calling an inaccessible method on an object.
* \_\_callStatic(): This magic method is invoked when calling an inaccessible static method on a class.
* \_\_toString(): This magic method is invoked when an object is converted to a string.
* \_\_invoke(): This magic method is invoked when an object is called as a function.

1. **Use of The $this keyword**

In PHP object-oriented programming (OOP), the keyword $this refers to the current instance of a class. It allows you to access the class's properties and methods within its own scope.