ADVANCE PHP

1. What Is Object Oriented Programming?
    - **object-oriented** is the combination of two words i.e. **object** and **oriented**. The dictionary meaning of the object is an article or entity that exists in the real world. The meaning of oriented is interested in a particular kind of thing or entity. In layman's terms, it is a programming pattern that rounds around an object or entity are called **object-oriented programming.**

2. **What Are Properties Of Object Oriented Systems?**
    - **Object:** Objects in Object Oriented Systems interact through messages.

    - **Inheritance:** The main class or the root class is called as a Base Class. Any class which is expected to have ALL properties of the base class along with its own is called as a Derived class. The process of deriving such a class is Derived class. For the “Food” class, a Derived class can be “Class Chinese Food”.

    - **Abstraction:** Abstraction is creating models or classes of some broad concept. Abstraction can be achieved through Inheritance or even Composition.

    - **Encapsulation:** Encapsulation is a collection of functions of a class and object. The “Food” class is an encapsulated form. It is achieved by specifying which class can use which members (private, public, protected) of an object.

    - **Polymorphism:** Polymorphism means existing in different forms. Inheritance is an example of Polymorphism. A base class exists in different forms as derived classes. Operator overloading is an example of Polymorphism in which an operator can be applied in different situations.

3. What Is the Difference Between Class And Interface?

Class: A class is a blueprint from which individual objects are created. A class can contain any of the following variable types.

- Local variables – Variables defined inside methods, constructors or blocks are called local variables. The variable will be declared and initialized within the method and the variable will be destroyed when the method has completed.
- Instance variables – Instance variables are variables within a class but outside any method. These variables are initialized when the class is instantiated. Instance variables can be accessed from inside any method, constructor or blocks of that particular class.
- Class variables – Class variables are variables declared within a class, outside any method, with the static keyword.

Interfaces

An interface is a reference type in Java. It is similar to class. It is a collection of abstract methods. A class implements an interface, thereby inheriting the abstract methods of the interface.

Along with abstract methods, an interface may also contain constants, default methods, static methods, and nested types. Method bodies exist only for default methods and static methods.

Writing an interface is similar to writing a class. But a class describes the attributes and behaviors of an object. And an interface contains behaviors that a class implements.

4. What Is Overloading?
    - Overloading refers to the ability to use a single identifier to define multiple methods of a class that differ in their input and output parameters. Overloaded methods are generally used when they conceptually execute the same task but with a slightly different set of parameters.

5. What Is T_PAAMAYIM_NEKUDOTAYIM (Scope Resolution Operator (::) with Example.

- The scope resolution operator is used to access variables and functions that exist within a specific scope. The scope resolution operator is very useful when two variables in different scopes have the same name.

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

- An abstract class is a class that is declared with the keyword `abstract` and may contain abstract and non-abstract methods. An abstract method is a method that is declared without an implementation. An abstract class cannot be instantiated, but can be subclassed by other classes that provide the implementation of the abstract methods. An abstract class is used to provide a common template or blueprint for subclasses that share some common behavior or attributes.

- An interface is a type that is declared with the keyword `interface` and may contain only abstract methods, constants, default methods, static methods, and private methods. An interface cannot be instantiated, but can be implemented by other classes that provide the implementation of all the abstract methods. An interface is used to specify a contract or a set of rules that the implementing classes must follow. An interface can also be used to achieve multiple inheritance in Java, as a class can implement more than one interface.

7. Define Constructor and Destructor?

    - A constructor is a special member function of a class that is used to initialize the objects of that class. A destructor is a special member function of a class that is used to destroy the objects of that class and release the resources they occupy. Both constructor and destructor have the same name as the class, but the destructor is preceded by a tilde (~) symbol. Constructors and destructors are automatically invoked when an object is created or deleted, respectively. They do not have any return type, not even void. Constructors can be overloaded, which means a class can have more than one constructor with different parameters. Destructors cannot be overloaded, which means a class can have only one destructor with no parameters. Constructors and destructors can be defined inside or outside the class, but they must have the scope resolution operator (::) if defined outside the class.

8. How to Load Classes in PHP?

    - One way to load classes in PHP is to use the autoloading feature, which allows you to register a function that will be called automatically when a class is used for the first time. This function can then include or require the file that contains the class definition, without the need to manually write the include or require statements for each class.

9. How to Call a Parent Constructor?

    - super keyword to call the parent constructor from the child constructor.

10. Are Parent Constructor Called Implicitly When Creating An ObjectOf Class?

    - A common way to do this is to use a keyword or a function that refers to the parent class, such as `super`, `parent`, or `base`, and then invoke the parent constructor with the appropriate arguments.

11. What Happens, If a Constructor Is Defined As Private Or Protected?

- A private or protected constructor also prevents the inheritance of the class by other classes. This is because a subclass needs to call the constructor of its superclass, either explicitly or implicitly, in its own constructor. If the superclass constructor is not accessible, the subclass cannot be created. To enable inheritance, a public or protected constructor must be provided.

12. What are PHP Magic Methods/Functions? List them Write programs for Static Keywords in PHP?

- PHP magic methods/functions are special methods that are called automatically when certain actions are performed on an object. They are prefixed with `__` (double underscore) to distinguish them from regular methods. They can be used to override PHP's default behavior or to provide some custom functionality. Some of the magic methods/functions in PHP are:
- `__construct()` and `__destruct()`: These are called when an object is created or destroyed, respectively. They can be used to initialize or clean up the object's properties or resources.

```
// Example of a static property and a static method
class Counter {
  public static $count = 0; // A static property

  public static function increment() { // A static method
    self::$count++; // Access the static property using self keyword
  }
}

echo Counter::$count; // Access the static property using :: operator
echo "<br>";
Counter::increment(); // Call the static method using :: operator
echo Counter::$count;
echo "<br>";

// Example of a static variable in a function
function add() {
  static $number = 0; // A static variable
  $number++;
  echo $number;
}

add(); // Call the function
echo "<br>";
add(); // Call the function again
echo "<br>";
add(); // Call the function again
```

Output :

```
0
1
2
1
2
3
```

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

To create multiple traits and use them in a single class, you can follow these steps:

- Declare the traits using the `trait` keyword, followed by the trait name and a block of code that contains the methods or properties you want to reuse.
- Declare the class using the `class` keyword, followed by the class name and a block of code that contains the class definition.
- Use the `use` keyword inside the class to import one or more traits, separated by commas. This will make the methods or properties of the traits available in the class.
- Optionally, you can also use the `as` operator to change the visibility or name of the imported methods or properties, or use the `instead of` operator to resolve any conflicts between the traits.

14. Write PHP Script of Object Iteration?

```
// Define a class with some properties and methods
class Book {
  public $title;
  public $author;
  public $price;
  private $rating;

  public function __construct($title, $author, $price, $rating) {
    $this->title = $title;
    $this->author = $author;
    $this->price = $price;
    $this->rating = $rating;
  }

  public function getRating() {
    return $this->rating;
  }
}
```

15. Use of The $this keyword

- The `$this` keyword is used to refer to the current object instance in PHP. It can be used to access the properties and methods of the object from within the object itself.

```
// Define a class named Person
class Person {
  // Declare a property named $name
  public $name;

  // Declare a constructor that takes a parameter
  public function __construct($name) {
    // Assign the parameter value to the $name property using $this
    $this->name = $name;
  }

  // Declare a method named sayHello
  public function sayHello() {
    // Print a greeting message using the $name property using $this
    echo "Hello, my name is " . $this->name . ".";
  }
}

// Create an object of the Person class with a name
$p = new Person("Alice");

// Call the sayHello method on the object
$p->sayHello();
```

16. Create Hotel Room Booking System User can book room by 3 ways • Full day • Half day • Custom

=>insert.php

```
<?php
// Insert a new reservation
$result = DB(
  "INSERT INTO `reservations` (`room_id`, `reservation_start`, `reservation_end`,
`reservation_name`, `reservation_email`) VALUES (?,?,?,?,?)",
```

```
  ["#01-A", "2023-12-01", "2023-12-03", "John Doe", "john@example.com"],
  "I"
);
// Check if successful
if ($result) {
  echo "Reservation ID: $result<br>";
} else {
  echo "Failed to insert reservation<br>";
}
?>
```

=> Update.php

```
<?php
// Update an existing reservation
$result = DB(
  "UPDATE `reservations` SET `reservation_end`=? WHERE `reservation_id`=?",
  ["2023-12-04", 1],
  "A"
);
// Check if successful
if ($result) {
  echo "Updated $result reservation(s)<br>";
} else {
  echo "Failed to update reservation<br>";
}
?>
```

=>Delete.php

```
<?php
// Delete an existing reservation
$result = DB(
  "DELETE FROM `reservations` WHERE `reservation_id`=?",
  [1],
  "A"
);
// Check if successful
if ($result) {
```

```
  echo "Deleted $result reservation(s)<br>";
} else {
  echo "Failed to delete reservation<br>";
}
?>

=>select.php

<?php
// Get the available rooms for a given date range
$start = "2023-12-01";
$end = "2023-12-03";
$available = DB(
  "SELECT * FROM `rooms` WHERE `room_id` NOT IN (SELECT `room_id` FROM
`reservations` WHERE `reservation_start`<=? AND `reservation_end`>=?)",
  [$end, $start]
);
// Loop through the available rooms and display them
foreach ($available as $a) {
  echo "Room: {$a['room_id']}<br>";
  echo "Type: {$a['room_type']}<br>";
  echo "Price: {$a['room_price']}<br>";
  echo "<hr>";
}
?>

<form id="booking-form" method="post" action="booking.php">
  <div>
    <label for="booking-type">Booking Type:</label>
    <select id="booking-type" name="booking-type" required>
      <option value="">Select</option>
      <option value="F">Full Day</option>
      <option value="H">Half Day</option>
      <option value="C">Custom</option>
    </select>
  </div>
  <div>
    <label for="room-type">Room Type:</label>
    <select id="room-type" name="room-type" required>
```

```
    <option value="">Select</option>
    <option value="S">Single</option>
    <option value="T">Twin</option>
    <option value="D">Double</option>
    <option value="B">Business</option>
    <option value="P">Presidential</option>
  </select>
</div>
<div>
  <label for="start-date">Start Date:</label>
  <input type="date" id="start-date" name="start-date" required>
</div>
<div>
  <label for="end-date">End Date:</label>
  <input type="date" id="end-date" name="end-date" required>
</div>
```

17. What is jQuery?

- jQuery is a popular JavaScript library that simplifies the client-side scripting of HTML. It allows you to perform tasks such as selecting and manipulating elements, creating animations, handling events, and making AJAX calls with less code and more efficiency. jQuery is also cross-browser compatible, meaning it works on different types of browsers without requiring much modification.

18. How are JavaScript and jQuery different?

    - JavaScript is a programming language that can run on web browsers and other platforms, while jQuery is a library of JavaScript code that simplifies common tasks such as selecting and manipulating elements, creating animations, handling events, and making AJAX calls.

19. Which is the starting point of code execution in jQuery?

    - The starting point of code execution in jQuery is the $ (document).ready () function, which is executed when the whole HTML DOM is loaded and is totally rendered by the browser1234. This function ensures that the event handlers and other jQuery methods work correctly without any errors. It

takes a callback function as an argument, which contains the jQuery code to be executed.

20. Document Load Vs Window. Load() jQuery

- The difference between document load and window load in jQuery is that the document load event is triggered when the HTML document is loaded and the DOM is ready, while the window load event is triggered when the entire page, including all the resources such as images, videos, and iframes, is fully loaded.

21. Image uploading with preview

```
<form id="upload-form" method="post" action="upload.php"
enctype="multipart/form-data">
  <input type="file" name="image" id="image" accept="image/*" required>
  <input type="submit" name="upload" id="upload" value="Upload">
</form>
<img id="preview" src="#" alt="Image Preview" width="300" height="300">

$(document).ready(function() {
  // Select the file input and the image preview
  var image = $("#image");
  var preview = $("#preview");

  // When the file input changes, update the image preview
  image.change(function() {
    // Get the selected file
    var file = this.files[0];

    // Check if the file is an image
    if (file && file.type.match("image.*")) {
      // Create a file reader
      var reader = new FileReader();

      // Set the image source to the reader result
      reader.onload = function(e) {
        preview.attr("src", e.target.result);
      };
```

```
      // Read the file as a data URL
      reader.readAsDataURL(file);
    } else {
      // Reset the image source
      preview.attr("src", "#");
    }
  });
});
```