**MODULE – 4(Advance PHP)**

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

**=>Ans::**->In Object Oriented Programming main program is divided into the small object depending on the problem.

->In OOPs data hinding possible.In OOP which prevent illegel access of function from outside of it.This is one of the best advantages of OOP.

->In OOPs Bottom up process is followed for proogram design.

->In OOPs importance is given to the data

->In OOPs mostly the data is private

->In OOPs there are public,private,protected specifiers

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

**=>Ans::**->Class

->object

->Class

->Objects

->Inheritance

->Interface

->Abstraction

Class:

->Class is collection of member function and instant variable

->Class is the blue print of the object

->class is same feature of the object

->Ex.Car class jaguar is the objects,wagnor objects of class Car

->color class red,blue

->class define by the class keyword

Object:

->An object is an instant of class

->It can be uniquely identified by its name and it drfines a state which is presented by the values of its attributes.

Inheritance:

->Inheritance is used for the inheriteance the properties of existing baseclass to the child class

->Inheritance is the process by which object of one class acquire the properties of object of another class.

->The concept of inheriteance provides the idea of reusability

Abstraction:

->Abstraction is the hinding the impementation and information

->Abstraction class at least one abstract method

Interface:

->We cannot create objects to interface, but the class implementing the interface can have objects

->We cannot define a variable in an interface.

->If we extend interface all the methods of the interface must be implemented in the child class.

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

**=>Ans::**=>A class describes the attributes and behaviors of an object.

->An interface contains behaviors that a class implements.

=>A class may contain abstract methods, concrete methods.

->An interface contains only abstract methods.

=>Members of a class can be public, private, protected or default.

->All the members of the interface are public by default.

**4. What Is Overloading?**

=>Creating two or more members that have the same name but are different in number or type of parameter is known as overloading

->Function overloading is the ability to create multiple functions of the same name with different implementations.

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

**=>Ans::**The scope resolution operator also known as *Paamayim Nekudotayim* or more commonly known as the double colon is a token that allows access to static, constant, and overridden properties or methods of a class.

->It is used to refer to blocks or codes in context to classes, objects, etc

**Example :** When an extending class overrides its parent’s function, the compiler calls the child class’s version of the method but it is up to the child class to call its parent’s version of the method.

<?php

class demo{

public function myfunc() {

echo "myfunc() of parent class\n ";

}

}

class child extends demo {

public function myfunc(){

// Calling parent's version

// of myfunc() method

parent::myfunc();

echo "myfunc() of child class";

}

}

$class = new child;

$class -> myfunc()

?>

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

**=>Ans::**

=>Abstract class can have abstract and non-abstract methods.

->Interface can have only abstract methods. Since Java 8, it can have default and static methods also.

=>Abstract class doesn't support multiple inheritance.

->Interface supports multiple inheritance.

=>The abstract keyword is used to declare abstract class.

->The interface keyword is used to declare interface.

=>An abstract class can be extended using keyword "extends".

->An interface can be implemented using keyword "implements".

=>)Example:

public abstract class Shape{

public abstract void draw();

}

-> public interface Drawable{

void draw();

}

**7. Define Constructor and Destructor?**

**=>Ans::**

Constructor:

=>Constructors are special class functions which performs initialization of every object.

->The Compiler calls the Constructor whenever an object is created.

->Constructors initialize values to object members after storage is allocated to the object.

Destructor :

->Destructor is a special class function which destroys the object as soon as the scope

of object ends.

->The destructor is called automatically by the compiler when the object

goes out of scope.

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

**=>Ans::**PHP load classes are used for declaring its object etc. in object oriented applications.

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

**=>Ans::**

<?php

class BaseClass {

function \_\_construct() {

print "In BaseClass constructor";

}

}

class SubClass extends BaseClass {

function \_\_construct() {

parent::\_\_construct();

print "In SubClass constructor";

}

}

class OtherSubClass extends BaseClass {

}

$obj = new BaseClass();

$obj = new SubClass();

$obj = new OtherSubClass();

?>

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

**=>Ans::**Parent constructors are not called implicitly if the child class defines a constructor.

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

<?php

class a

{

function \_\_construct()

{

echo "Hi";

}

}

$obj = new a;

?>

Output=>Give error

**12. What are PHP Magic Methods/Functions? List the**

**=>Ans::** =>The following method names are considered magical: \_\_construct(), \_\_destruct(), \_\_call(), \_\_callStatic(), \_\_get(), \_\_set(), \_\_isset(), \_\_unset(), \_\_sleep(), \_\_wakeup(), \_\_serialize(), \_\_unserialize(), \_\_toString(), \_\_invoke(), \_\_set\_state(), \_\_clone(), and \_\_debugInfo().

Magic methods are special methods which override PHP's default's action when certain actions are performed on an object.

**13. Write program for Static Keyword in PHP?**

**=>Ans::**<?php

class A

{

function show()

{

echo "This is A Class show function";

}

}

class B

{

static function show()

{

echo "This is B Class show function";

}

}

$obja = new A;

$obja->show();

B::show();

?>

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

**=>Ans::**

<?php

trait message1 {

public function msg1() {

echo "OOP is fun! ";

}

}

trait message2 {

public function msg2() {

echo "OOP is fun too! ";

}

}

class Welcome {

use message1;

use message2;

}

$obj = new Welcome();

$obj->msg1();

$obj->msg2();

?>

**15.Write PHP Script of Object Iteration?**

**=>Ans::** PHP provides a way for objects to be defined so it is possible to iterate through a list of items,with for example a foreach statement.

<?php  
class MyClass  
{  
    public $var1 = 'value 1';  
    public $var2 = 'value 2';  
    public $var3 = 'value 3';  
  
    protected $protected = 'protected var';  
    private   $private   = 'private var';  
  
    function iterateVisible() {  
       echo "MyClass::iterateVisible:\n";  
       foreach ($this as $key => $value) {  
           print "$key => $value\n";  
       }  
    }  
}  
  
$class = new MyClass();  
  
foreach($class as $key => $value) {  
    print "$key => $value\n";  
}  
echo "\n";  
  
  
$class->iterateVisible();  
  
?>

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

**=>Ans::**

=>In PHP, $this keyword references the current object of the class.

->The $this keyword allows you to access the properties and methods of the current object within the class using the object operator (->)

->The $this keyword is only available within a class. It doesn’t exist outside of the class. If you attempt to use the $this outside of a class, you’ll get an error.

EX:

<?php

class A

{

public $x=50;

function demo($x)

{

echo $x."<br/>";//output::10

echo $this->x;//this keyword thi function ma aapeli value use thay output::50

}

}

$obj = new A;

$obj->demo(10);

?>