OOP IN PHP

Viva Vice on OOP in PHP

Q1. What is OOPS?

OOPS is abbreviated as Object Oriented Programming system in which programs are considered as a collection of objects. Each object is nothing but an instance of a class.

Q2. Write basic concepts of OOPS?

Classes.

Objects.

Abstraction.

Encapsulation.

Inheritance.

Polymorphism.

Q3. What is a class?

A class is simply a representation of a type of object. It is the blueprint/ plan/ template that describe the details of an object.

Q4. What is an object?

Object is termed as an instance of a class, and it has its own state, behavior and identity.

Q5. What is Encapsulation?

Wrapping up member variables and methods together into a single unit (i.e. Class) is called Encapsulation.

Q6. What is Polymorphism?

It is simply “One thing, can use in different forms”

or example, One car (class) can extend two classes (Honda & Hyundai)

Q7. . What is Inheritance?

Inheritance is a concept where one class shares the structure and behavior defined in

another class.

Q8. Which version of PHP supports OOPs concepts?

PHP5

Q9. Write the syntax of declaring Class in PHP?

```
<?php
class ClassName
   {
      // Class properties and methods
   }
?>
```

Q10. Write the syntax Of declaring object in PHP?

```
$obj = new MyClass();
```

Q11. What Is Difference Between Class And Interface?

Interfaces do not contain business logic

You must extend interface to use.

You can't create object of interface.

Q12. What Is Static Keyword In Php?

If we declare a Method or Class Property as static, then we can access that without use of instantiation of the class. Static Method are faster than Normal method.

Q13. What Is Object Iteration?

PHP provides a way for objects to be iterate through a list of items, for this we can use foreach. Only visible properties will be listed.

Q14. Difference between class and an object?

An object is an instance of a class. Objects hold any information , but classes don't have any information. Definition of properties and functions can be done at class and can be used by the object.

Class can have sub-classes, and an object doesn't have sub-objects.

Q15. What is the difference between structure and a class?

A structure is used for grouping data whereas class can be used for grouping data and methods. Structures are exclusively used for data and it doesn't require strict validation , but classes are used to encapsulates and inherit data which requires strict validation.

Q16. What are the advantages of object oriented programming?

Code Reusability: it can be achieved through inheritance and traits.

Modularity: it can be achieved through breaking large code into small modules, Modularity reduces complexity.

Flexibility: it can be achieved through polymorphism.

Maintainability: it is to maintain code which follows Object Oriented Programming Concepts.

Security: it can be achieved through Encapsulation.

Testability: it is easy to test.

References:

1. Sebesta,"Concept of programming Language", Pearson Edu
2. Louden, "Programming Languages: Principles & Practices" , Cengage Learning
3. Tucker, "Programming Languages: Principles and paradigms ", Tata McGraw –Hill.
4. E Horowitz, "Programming Languages", 2nd Edition, Addison Wesley

Related Posts:

1. Sequence Control & Expression | PPL
2. PPL:Named Constants
3. Parse Tree | PPL | Prof. Jayesh Umre
4. Basic elements of Prolog
5. Loops | PPL | Prof. Jayesh Umre

6. Subprograms Parameter passing methods | PPL | Prof. Jayesh Umre
7. Programming Paradigms | PPL | Prof. Jayesh Umre
8. Subprograms Introduction | PPL | Prof. Jayesh Umre
9. Phases of Compiler | PPL | Prof. Jayesh Umre
10. Parse Tree | PPL
11. Influences on Language design | PPL | Prof. Jayesh Umre
12. Fundamentals of Subprograms | PPL | Prof. Jayesh Umre
13. Programming Paradigm
14. Influences on Language Design
15. Language Evaluation Criteria
16. OOP in C++ | PPL
17. OOP in C# | PPL
18. OOP in Java | PPL
19. PPL: Abstraction & Encapsulation
20. PPL: Semaphores
21. PPL: Introduction to 4GL
22. PPL: Variable Initialization
23. PPL: Conditional Statements
24. PPL: Array
25. PPL: Strong Typing
26. PPL: Coroutines
27. PPL: Exception Handler in C++
28. PPL: Character Data Type
29. PPL: Exceptions
30. PPL: Heap based storage management
31. PPL: Primitive Data Type
32. PPL: Data types

33. Programming Environments | PPL
34. Virtual Machine | PPL
35. PPL: Local referencing environments
36. Generic Subprograms
37. Local referencing environments | PPL | Prof. Jayesh Umre
38. Generic Subprograms | PPL | Prof. Jayesh Umre
39. PPL: Java Threads
40. PPL: Loops
41. PPL: Exception Handling
42. PPL: C# Threads
43. Pointer & Reference Type | PPL
44. Scope and lifetime of variable
45. Design issues for functions
46. Parameter passing methods
47. Fundamentals of sub-programs
48. Subprograms
49. Design issues of subprogram
50. Garbage Collection
51. Issues in Language Translation
52. PPL Previous years solved papers
53. Type Checking | PPL | Prof. Jayesh Umre
54. PPL RGPV May 2018 solved paper discussion| Prof. Jayesh Umre
55. PPL Viva Voce
56. PPL RGPV June 2017 Solved paper | Prof. Jayesh Umre
57. Concurrency
58. Basic elements of Prolog
59. Introduction and overview of Logic programming

60. Application of Logic programming
61. PPL: Influences on Language Design
62. Language Evaluation Criteria PPL
63. PPL: Sequence Control & Expression
64. PPL: Programming Environments
65. PPL: Virtual Machine
66. PPL: Programming Paradigm
67. PPL: Pointer & Reference Type
68. try-catch block in C++