<?php

/\*

From PHP5, you can also write PHP code in an object-oriented style.

OOP stands for Object-Oriented Programming.

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.

Object-oriented programming has several

advantages over procedural programming:

OOP is faster and easier to execute

OOP provides a clear structure for the programs

OOP provide securities

OOP helps to keep the PHP code DRY "Don't Repeat Yourself", and makes the

code easier to maintain, modify and debug

OOP makes it possible to create full reusable applications

with less code and shorter development time

Example :

Class === Fruits / car

Object ==== apple,banana,orange / volve audi ford

oops Features :

Class

Object

Encapsulation

Inheritance / Reusability == extends

Access Modifier / Visibility public private protected

Polymorphism / overloading / overriding

Class :

Define a Class

making group of data member(variable) and member function that called class

A class is defined by using the class keyword, followed by the name of the class and a pair of curly braces ({}). All its properties and methods go inside the braces:

Define Objects

Classes are nothing without objects! We can create multiple objects from a class. E

ach object has all the properties and methods defined in the class, but they will have different property values.

Objects of a class are created using the new keyword.

\*/

/\*

$a=10;

$b=20;

echo $a+$b;

function simple()

{

$a=10;

$b=20;

echo $a+$b;

}

simple();

\*/

/\*

class abc

{

function simple()

{

$a=10;

$b=20;

echo $a+$b;

}

}

$obj=new abc;

$obj->simple();

\*/

/\*

class abc

{

function sum()

{

$a=10;$b=20; // local var only access in function

echo $a+$b;

}

function multi()

{

$a=10;$b=20;

echo $a\*$b;

}

}

$obj=new abc;

$obj->sum();

$obj->multi();

\*/

class abc

{

public $a=10; // aaccess anywhere in class by $this-> keywords

public $b=20;

function sum()

{

echo $this->a+$this->b;

}

function multi()

{

$this->sum();

echo $this->a\*$this->b;

}

}

$obj=new abc;

$obj->multi();