PHP

a)PHP is a server scripting language, and is a powerful tool for making dynamic and interactive Web pages quickly.

b)PHP is a widely-used, free, and efficient alternative to competitors such as Microsoft's ASP.

c) PHP started out as a small open source project that evolved as more and more people found out how useful it was. Rasmus Lerdorf unleashed the first version of PHP way back in 1994.

d) PHP is an acronym for "PHP Hypertext Preprocessor",PHP costs nothing, it is free to download and use.

What is a PHP File?

i)PHP files can contain text, HTML, CSS, JavaScript, and PHP code

ii)PHP code are executed on the server, and the result is returned to the browser as plain HTML

iii)PHP files have extension ".php".

What Can PHP Do?

i)PHP can generate dynamic page content

ii)PHP can create, open, read, write, delete, and close files on the server

iii)PHP can collect form data

v)PHP can send and receive cookies

v)PHP can add, delete, modify data in your database

vi)PHP can restrict users to access some pages on your website

vii)PHP can encrypt data

viii)With PHP you are not limited to output HTML. You can output images, PDF files, and even Flash movies. You can also output any text, such as XHTML and XML.

- **Why PHP?**
- i)PHP runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
- ii)PHP is compatible with almost all servers used today (Apache, IIS, etc.)
- iii)PHP supports a wide range of databases
- iv)PHP is easy to learn and runs efficiently on the server side

Characteristics of PHP

Five important characteristics make PHP's practical nature possible:

i)Simplicity

ii) Efficiency

iii)Security

iv)Flexibility

v)Familiarity.

PHP Installation:-

a)To run php you required , you must:

i)install a web server

ii)install PHP:-

iii)install a database, such as MySQL

PHP Syntax:-

The PHP script is executed on the server, and the plain HTML result is sent back to the browser.

a)A PHP script can be placed anywhere in the document.

b)A PHP script starts with **<?php** and ends with **?>**:

```
<?php
    // PHP code goes here
    ?>
```

c)A PHP file normally contains HTML tags, and some PHP scripting code.

e)Example

```
<!DOCTYPE html>
    <html> <body>
    <h1>My first PHP page</h1>
    <?php
    echo "Hello World!";
    ?>
    </body>  </html>
```

- **Comments in PHP**
- A comment in PHP code is a line that is not read/executed as part of the program. Its only purpose is to be read by someone who is editing the code!.
- **Example**
-
```
<!DOCTYPE html>
<html> <body>
<?php
// This is a single line comment
# This is also a single line comment
/*
This is a multiple lines comment block
that spans over more than
one line
*/
?>  </body>  </html>
```

- **PHP Case Sensitivity**
- i)In PHP, all user-defined **functions, classes, and keywords** (e.g. if, else, while, echo, etc.) are **NOT case-sensitive.**
- ii)In the example below, all three echo statements below are legal (and equal):
- **iii)Example**
-
```
<html> <body>
<?php
ECHO "Hello World!<br>";
echo "Hello World!<br>";
EcHo "Hello World!<br>";
?>
```

- in PHP, all **variables are case-sensitive**.
- In the example below, only the first statement will display the value of the $color variable (this is because $color, $COLOR, and $coLOR are treated as three different variables):
- **Example**
-
  ```
  <!DOCTYPE html>
  <html> <body>
  <?php
  $color="red";
  echo "My car is " . $color . "<br>";
  echo "My house is " . $COLOR . "<br>";
  echo "My boat is " . $coLOR . "<br>";
  ?></body></html>
  ```

PHP Variables:-

a)PHP variables can be used to hold values (x=5) or expressions (z=x+y).

b)A variable can have a short name (like x and y) or a more descriptive name (age, carname, total_volume).

c)Rules for PHP variables:

A variable starts with the $ sign, followed by the name of the variable

A variable name must start with a letter or the underscore character

A variable name cannot start with a number

A variable name can only contain alpha-numeric characters and underscores(A-z,0-9,and _)

Variable names are case sensitive ($y and $Y are two different variables).

d)**Creating (Declaring) PHP Variables**

PHP has no command for declaring a variable.A variable is created the moment you first assign a value to it:

Example

```
<?php
    $txt="Hello world!";
    $x=5;
    $y=10.5;
    ?>
```

echo and print Statements

a)In PHP there are two basic ways to get output: echo and print.

b)There are some differences between echo and print:

echo - can output one or more strings

print - can only output one string, and returns always 1.

echo is marginally faster compared to print as echo does not return any value.

c) The PHP echo Statement

echo is a language construct, and can be used with or without parentheses: echo or echo().

Display Strings

The following example shows how to display different strings with the echo command (also notice that the strings can contain HTML markup):

Example

```
<?php
    echo "<h2>PHP is fun!</h2>";
    echo "Hello world!<br>";
    echo "I'm about to learn PHP!<br>";
    echo "This", " string", " was", " made", " with multiple parameters.";
    ?>
```

Output:- PHP is fun!

```
Hello world!
    I'm about to learn PHP!
    This string was made with multiple parameters.
```

If we run above code using print statement it will give an error on line no 5.because print statement take only one parameter.

d) The print Statement

print is also a language construct, and can be used with or without parentheses: print or print().

Display Strings

The following example shows how to display different strings with the print command (also notice that the strings can contain HTML markup):

Example

```
<?php
    print "<h2>PHP is fun!</h2>";
    print "Hello world!<br>";
    print "I'm about to learn PHP!";
    ?>
```

Output:- PHP is fun!

```
Hello world!
    I'm about to learn PHP!
```

PHP 5 Data Types

a)PHP Strings

```
<?php
      $x = "Hello world!";
      echo $x;
      ?>
```

b) Integers

```
<?php
      $x = 5985;
      var_dump($x);   echo "<br>";
      $x = -345; // negative number
      var_dump($x);   echo "<br>";
      $x = 0x8C; // hexadecimal number
      var_dump($x);   echo "<br>";
      $x = 047; // octal number
      var_dump($x);
      ?>
```

- Output:-
```
int(5985)
int(-345)
int(140)
int(39).
```

PHP Arrays

An array stores multiple values in one single variable.

In the following example we create an array, and then use the PHP var_dump() function to return the data type and value of the array:

Example

```
<?php
$cars=array("Volvo","BMW","Toyota");
var_dump($cars);
?>
```

Output:- array(3) { [0]=> string(5) "Volvo" [1]=> string(3) "BMW" [2]=> string(6) "Toyota" }

PHP NULL Value

The special NULL value represents that a variable has no value. NULL is the only possible value of data type NULL.

```
<?php
$x="Hello world!";
$x=null;
var_dump($x);
?> output:- NULL
```

PHP Objects

An object is a data type which stores data and information on how to process that data.

In PHP, an object must be explicitly declared.

First we must declare a class of object. For this, we use the class keyword. A class is a structure that can contain properties and methods.

We then define the data type in the object class, and then we use the data type in instances of that class:

```
<!DOCTYPE html><html><body>
<?php
class Car
{    var $color;
    function Car($color) {
      $this->color = $color;
    }
    function what_color() {
    echo $this->color;
    }
}
// instantiate one object
$a = new Car("white");
$a-> what_color();
?>  </body></html>
```

PHP String Functions

Function name	Explation	Function Code
str_ireplace()	Replaces some characters in a string (case-insensitive)	<?php echo str_ireplace("WORLD","Peter","Hello world!"); ?> output:- Hello Peter!
str_pad()	Pads a string to a new length	<?php $str = "Hello World"; echo str_pad($str,20,"."); ?> Output:- Hello World.........
str_repeat()	Repeats a string a specified number of times	<?php echo str_repeat(".",13); ?> Output:- ....
str_replace()	Replaces some characters in a string (case-sensitive)	<?php echo str_replace("world","Peter","Hello world!"); ?> Output:-Hello Peter
str_rot13()	Performs the ROT13 encoding on a string	<?php echo str_rot13("Hello World"); ?> Outout:- Uryyb Jbeyq

str_shuffle()	Randomly shuffles all characters in a string	`<?php`
		`echo str_shuffle("Hello World");`
		`?>` Output:- drloWHeol l

str_split()	Splits a string into an array	`<?php`
		`print_r(str_split("Hello"));`
		`?>`
		Output:- Array ([0] => H [1] => e [2] => l [3] => l [4] => o)

str_word_count()	Count the number of words in a string	`<?php`
		`echo str_word_count("Hello world!");`
		`?>` output:-2

strcasecmp()	Compares two strings (case-insensitive)	`<?php`
		`echo strcasecmp("Hello world!","HELLO WORLD!");`
		`?>` Output:-0

strchr()	Finds the first occurrence of a string inside another string (alias of strstr())	`<?php`
		`echo strchr("Hello world!","world");`
		`?>` Output:- world!

strcmp()	Compares two strings (case-sensitive)	<?php echo strcmp("Hello world!","Hello world!"); ?> output=0
strcspn()	Returns the number of characters found in a string before any part of some specified characters are found	<?php echo strcspn("Hello world!","w"); ?> Output:- 6
stripos()	Returns the position of the first occurrence of a string inside another string (case-insensitive)	<?php echo stripos("I love php, I love php too!","PHP"); ?> Output:- 7
stristr()	Finds the first occurrence of a string inside another string (case-insensitive)	<?php echo stristr("Hello world!","WORLD"); ?> Output:- world!
strlen()	Returns the length of a string	<?php echo strlen("Hello"); ?> Output:-5
strpos()	Returns the position of the first occurrence of a string inside another string (case-sensitive)	<?php echo strpos("I love php, I love php too!","php"); ?> Output:- 7
strrchr()	Finds the last occurrence of a string inside another string	<?php echo strrchr("Hello world!","world"); ?> Output:- world!

strrev()	Reverses a string	<?php echo strrev("Hello World!"); ?> Output:- !dlroW olleH
strtolower()	Converts a string to lowercase letters	<?php echo strtolower("Hello WORLD."); ?> Output:- hello world.
strtoupper()	Converts a string to uppercase letters	<?php echo strtoupper("Hello WORLD!"); ?> Output:- HELLO WORLD!
strtr()	Translates **certain** characters in a string	<?php echo strtr("Hilla Warld","ia","eo"); ?> Output:- Hello World
substr()	Returns a part of a string	<?php echo substr("Hello world",6); ?> Output:- world
ucfirst()	Converts the first character of a string to uppercase	<?php echo ucfirst("hello world!"); ?> Output:- Hello world!
ucwords()	Converts the first character of each word in a string to uppercase	<?php echo ucwords("hello world"); ?> Output:- Hello World

Statement	Description	Example
if...else statement	use this statement if you want to execute a set of code when a condition is true and another if the condition is not true	<html><body><?php $d=date("D");if ($d=="Fri") echo "Have a nice weekend!"; else echo "Have a nice day!"; ?></body></html>
elseif statement	is used with the if...else statement to execute a set of code if **one** of several condition are true	<html><body><?php $d=date("D");if ($d=="Fri") echo "Have a nice weekend!"; elseif ($d=="Sun") echo "Have a nice Sunday!"; else echo "Have a nice day!"; ?></body></html>
switch statement	is used if you want to select one of many blocks of code to be executed, use the Switch statement. The switch statement is used to avoid long blocks of if..elseif..else code.	<html><body><?php $d=date("D"); switch ($d) {case "Mon": echo "Today is Monday"; break;case "Tue": echo "Today is Tuesday"; break; default: echo "Wonder which day is this“ ;}?></body></html>

for	loops through a block of code a specified number of times.	<html><body> <?php $a = 0; $b = 0; for($i=0; $i<5; $i++) { $a += 10; $b += 5; } echo ("At the end of the loop a=$a and b=$b"); ?></body></html>
while	loops through a block of code if and as long as a specified condition is true.	<html><body> <?php $i = 0; $num = 50; while($i < 10) { $num--; $i++; } echo ("Loop stopped at i = $i and num = $num"); ?> </body></html>

do…while	The do...while statement will execute a block of code at least once - it then will repeat the loop as long as a condition is true.	`<html><body>`
		`<?php`
		`$i = 0;`
		`$num = 0;`
		`do`
		`{  $i++;`
		`}while( $i < 10 );`
		`echo ("Loop stopped at i = $i" );`
		`?>`
		`</body></html>`

foreach	The foreach loop works only on arrays, and is used to loop through each key/value pair in an array.	`<html><body>`
		`<?php`
		`$array = array( 1, 2, 3, 4, 5);`
		`foreach( $array as $value )`
		`{`
		`  echo "Value is $value <br />";`
		`}`
		`?></body></html>`

- **An array** :-is a data structure that stores one or more similar type of values in a single value. For example if you want to store 100 numbers then instead of defining 100 variables its easy to define an array of 100 length.
- There are three different kind of arrays and each array value is accessed using an ID c which is called array index.
- **Numeric array** - An array with a numeric index. Values are stored and accessed in linear fashion also called **Indexed arrays.**
- **Associative array** - An array with strings as index. This stores element values in association with key values rather than in a strict linear index order.
- **Multidimensional array** - An array containing one or more arrays and values are accessed using multiple indices

a)Numeric array:

i)These arrays can store numbers, strings and any object but their index will be prepresented by numbers. By default array index starts from zero.

Ii)Example:Here we have used **array()** function to create array.

```
<html><body>
<?php
$numbers = array( 1, 2, 3, 4, 5);
/* First method to create array. */
foreach( $numbers as $value )
{  echo "Value is $value <br />";
}$numbers[0] = "one";
 /* Second method to create array. */
$numbers[1] = "two";
$numbers[2] = "three";
$numbers[3] = "four";
$numbers[4] = "five";
foreach( $numbers as $value )
{  echo "Value is $value <br />";}?>
</body>
</html>
```

Value is 1
Value is 2
Value is 3
Value is 4
Value is 5
Value is one
Value is two
Value is three
Value is four
Value is five

- **b) Associative Arrays**

i)The associative arrays are very similar to numeric arrays in term of functionality but they are different in terms of their index. Associative array will have their index as string so that you can establish a strong association between key and values.

```
<?php
$salaries = array( "mohammad" => 2000, "qadir" => 1000,
    "zara" => 500    );
echo "Salary of mohammad is ". $salaries['mohammad'] .
    "<br />";
echo "Salary of qadir is ".  $salaries['qadir']. "<br />";
Echo"Salary of zara is ".  $salaries['zara']. "<br />";
/* Second method to create array. */
$salaries['mohammad'] = "high";
$salaries['qadir'] = "medium";
$salaries['zara'] = "low";
echo "Salary of mohammad is ". $salaries['mohammad'] .
    "<br />";echo "Salary of qadir is ".  $salaries['qadir'].
    "<br />";echo "Salary of zara is ".  $salaries['zara']. "<br
    />";
?>
```

- **Loop Through an Associative Array**
- To loop through and print all the values of an associative array, you could use a foreach loop, like this:
- **Example**
-
```
<?php
$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"4
3");
foreach($age as $x=>$x_value) {
  echo "Key=" . $x . ", Value=" . $x_value;
  echo "<br>";
}
?>
```
- Output:-
```
Key=Peter, Value=35
Key=Ben, Value=37
Key=Joe, Value=43
```

- **Multidimensional Arrays**
- A multi-dimensional array each element in the main array can also be an array. And each element in the sub-array can be an array, and so on. Values in the multi-dimensional array are accessed using multiple index.

```
<?php
  $marks = array(
        "mohammad" => array
        ("physics" => 35,
        "maths" => 30,
        "chemistry" => 39
        ),
        "qadir" => array
             ( "physics" => 30,
             "maths" => 32,
             "chemistry" => 29
             ),
             "zara" => array
             ( "physics" => 31,
             "maths" => 22,
             "chemistry" => 39
             )
          );
  /* Accessing multi-dimensional array values */
  echo "Marks for mohammad in physics : " ;
  echo $marks['mohammad']['physics'] . "<br />";
  echo "Marks for qadir in maths : ";
  echo $marks['qadir']['maths'] . "<br />";
  echo "Marks for zara in chemistry : " ;
  echo $marks['zara']['chemistry'] . "<br />";
?>
```

Function name	Explanation	Function Code
array_change_key_case()	Changes all keys in an array to lowercase or uppercase	<?php $age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43"); print_r(array_change_key_case($age,CASE_UPPER)); ?> output:- Array ([PETER] => 35 [BEN] => 37 [JOE] => 43)

array_column()	Returns the values from a single column in the input array	<?php
		$a = array(
		  array(
		    'id' => 5698, 'first_name' => 'Peter',
		    'last_name' => 'Griffin',),
		  array(
		    'id' => 4767, 'first_name' => 'Ben', 'last_name' => 'Smith',)
		);
		$last_names = array_column($a, 'last_name');
		print_r($last_names);
		?>
		Output:- Array
		(
		  [0] => Griffin
		  [1] => Smith
		)

array_combine()	Creates an array by using the elements from one "keys" array and one "values" array	<?php $fname=array("Peter","Ben","Joe"); $age=array("35","37","43"); $c=array_combine($fname,$age); print_r($c); ?> Output:- Array ([Peter] => 35 [Ben] => 37 [Joe] => 43)
array_count_values()	Counts all the values of an array	<?php $a=array("A","Cat","Dog","A","Dog"); print_r(array_count_values($a)); ?>output:- Array ([A] => 2 [Cat] => 1 [Dog] => 2)
array_diff()	Compare arrays, and returns the differences (compare values only)	<?php $a1=array("a"=>"red","b"=>"green","c"=>"blue","d"=>"yellow"); $a2=array("e"=>"red","f"=>"black","g"=>"purple"); $a3=array("a"=>"red","b"=>"black","h"=>"yellow"); $result=array_diff($a1,$a2,$a3); print_r($result); ?>output:- Array ([b] => green [c] => blue)

array_key_exists()	Checks if the specified key exists in the array	<?php
		$a=array("Volvo"=>"XC90","BMW"=>"X5");
		if (array_key_exists("Volvo",$a))
		 { echo "Key exists!";
		 }
		else
		 { echo "Key does not exist!";
		 }
		?> Output:- Key exists!

array_keys()	Returns all the keys of an array	<?php $a=array("Volvo"=>"XC90","BMW"=>"X5","Toyota"=>"Highlander"); print_r(array_keys($a)); ?> output:- Array ([0] => Volvo [1] => BMW [2] => Toyota)
array_merge()	Merges one or more arrays into one array	<?php $a1=array("red","green"); $a2=array("blue","yellow"); print_r(array_merge($a1,$a2)); ?> output:- Array ([0] => red [1] => green [2] => blue [3] => yellow)

array_multisort()	Sorts multiple or multi-dimensional arrays	<?php
		$a1=array("Dog","Dog","Cat");
		$a2=array("Pluto","Fido","Missy");
		array_multisort($a1,SORT_ASC,$a2,SORT_DESC);
		print_r($a1);
		print_r($a2);
		?>
		Output:- Array ([0] => Cat [1] => Dog [2] => Dog)
		Array ([0] => Missy [1] => Pluto [2] => Fido)

array_pad()	Inserts a specified number of items, with a specified value, to an array	<?php $a=array("red","green"); print_r(array_pad($a,5,"blue")); ?> output:- Array ([0] => red [1] => green [2] => blue [3] => blue [4] => blue)

array_pop()	Deletes the last element of an array	<?php
		$a=array("red","green","blue");
		array_pop($a);
		print_r($a);
		?> output:- Array ([0] => red [1] => green)