PHP

UNIT - 1

Chapter-2 : PHP Basic

Prepared By : Ms. Kakadiya Jainam A.

Basic Introduction to PHP

- The full form of php is **"Hypertext Preprocessor".** Its original name is **"Personal Home Page".**
- PHP is a powerful server-side scripting language for creating dynamic and interactive webpages.
- The PHP syntax is very similar to **Perl** and **C** programming.
- PHP is often used together with Apache (web server) on various operating systems.
- A PHP file may contain text, HTML tags and scripts.

Conn…

- Scripts in a PHP file are executed on the server.
- Web page is a web document which consists of text, graphics and media.
- Any webpage is created with HTML tags or any markup language.
- Webpage can read by any browser.
- Basically webpage with formatting language have different extension like .HTML, .HTM etc.

History of Php

- History
  - History of PHP is as given.

Versions	Released Date	Description
1	1995	It was the first version which is referred to as Personal Home Page introduced by Rasmus Ledorf.

CONN..

- History

Versions	Released Date	Description
2	1997	The newer version of PHP was used over by 50,000 websites for creating dynamic web pages.

CONN..

- History

Versions	Released Date	Description
3	1998	Then after two scientists **Zeev Suraski** and **Andi Gutmans** introduced some extra features into the older version. So that it become faster and simple tool for creating web-sites.

CONN..

- History

Versions	Released Date	Description
4	2000	When PHP's 4th version was released, the concept of **Zend Engine** was introduced, **Super global variables** like **$_GET**, **$_POST**, **$_SESSION** were introduced, more security was provided.

CONN..

- History

Versions	Released Date	Description
5	2004	The newer version introduced the **Zend Engine II** with the object oriented features, enabled the filter extension, many bugs were fixed.

Why PHP?

- There are various reasons which are listed…
  - Database Connectivity
    - PHP supports many databases Like, MySQL, Informix, Oracle, Sybase, Solid, PostgreSQL, Generic ODBC, etc.
  - Free Ware
    - PHP and MYSQL are freeware which means that they can be easily downloaded without any cost.
    - It is an open source software. So no charge is to be paid for it.
    - PHP is FREE to download from the official PHP resource: www.php.net

CONN..

- Easy to Use
  - Compared to other languages it is easy to use as the coding of PHP are similar to that of C language and it is not hard to learn.
- Speed
  - Compared to other heavy software like, .Net & Java, which are used to develop websites the speed required to open the webpage create in PHP is much faster.

CONN..

- Compatible with web-server
  - PHP code is processed by different WebServer.
  - Generally, there are three types of Web Server in use, they are
    - Apache,
    - IIS (Internet Information Services) and
    - Netscape Enterprise Service.
    - PHP can execute on any of the above WebServer, but Apache is the webserver which is used widely with PHP

CONN..

- Cross platform
  - CrossPlatform interoperability means operating system independent and machine independent.
  - PHP can work on any Operating System starting from low level like UNIX to high level like Windows-8.
- HTML Support
  - PHP is a scripting language and we can also use HTML tags in side of PHP file.
  - PHP uses various controls from html.

CONN..

- Stability
  - Compared to other language like JSP where the server is required to be rebooted each time when any modification is made in the web-page.
  - For PHP there is no issue of rebooting the server.
  - For PHP, the stability comes by two ways:
    - The WebServer is not required to be rebooted again and again.
    - If the subversion of PHP and MYSQL are changed then also they are compatible.

What is a PHP File?

- PHP files may contain text, HTML tags and scripts
- PHP files are returned to the browser as plain HTML page.
- PHP files have a file extensions like,
  - ".php“ (Mostly used with php file)
  - ".php3" or
  - ".phtml"

What is MySQL?

- MySQL is a database server
- MySQL is ideal for both small and large applications
- MySQL supports standard SQL
- MySQL compiles on a number of platforms
- MySQL is free to download and use

PHP + MySQL

- PHP combined with MySQL are cross-platform (means that we can develop web-pages using Windows and serve on a Unix or any other platform.)

UNDERSTANDING OF PHP.INI & PHP .HT ACCESS FILE

- The PHP configuration file, php.ini, is the final and most immediate way to affect PHP's functionality.
- The php.ini file is read each time PHP is initialized.in other words.
- htaccess is a configuration file for use on web servers running the Apache Web Server software.
- When a .htaccess file is placed in a directory which is in turn 'loaded via the Apache Web Server', then the .htaccess file is detected and executed by the Apache Web Server software.

How TO execute PHP Codes…

- To Execute PHP Codes…

  - Start WAMP server
  - Open Internet Explorer or any other web-browser.
  - Type the URL as
    - http://localhost
  - Now Press ENTER & it will display the index page…
  - Type Name of your PHP file after ***localhost*** and execute it…

VARIABLES

- Computer has memory cells same as human mind to remembering the values.
- Each memory cells in the computer has unique address which will be in binary format.
- It is not possible to remember the address of each memory cell so that cell is introduced with a name which is called "**variable".**
- Variable is a storage area where the values can be stored. These values may be a fixed value or the value is inputted by the user.

How to define Variables in PHP?

Syntax : $variable_name=value

Purpose : To define variable in PHP we can use above syntax.

Example :

```
<?php
        $a=10;
        $b=20;
        $c=$a+$b;
        echo $c;
?>
```

RULES FOR VARIABLE NAMING:

- First letter of variable name must be an alphabet.
- The variable name can consist of alphabets and digits.
- There is no space is allowed between the variable name.
- No special symbol other then **underscore(_)** is used.
- We can not use Keyword as variable name.
- We can use maximum 31 characters as length of variable name.

Data Types

- The variable which are defined in PHP has some values.
- In PHP the data type is not specified while declaring a variable.
- But still there may be situation when it is required to identify the data type.
- We can use Primary data-types as given
  - Integer Types
  - Floating Point Types
  - String Types
  - Boolean Types.

Integer Types

- Integer has the memory size of 16bit which means 2 Bytes.
- Integer data type has a fixed range of
  - -32,768 to +32767
- If the value will cross the limit in positive or negative range then it will return the **garbage values.**
- Example:

```
<?php
        $a=250;
?>
```

Floating Point Types (Double)

- Floating has the memory size of 32bit which means 4 Bytes.
- Floating point data type has a fixed range of
  - 3.4e-38 to 3.4e+38. **(3.4*10^38)**
- If the value will cross the limit in positive or negative range then it will return the **garbage values.**
- Example:

```
<?php
        $percent=62.66666;
?>
```

String Type

- When there are combination of more than one character or digit or any special character is to be used then it comes in the category of string.

- A string is mixture of each character present in the keyboard.

- **Example:**

```
<?php
    $name="Monarch";
    echo "<b>",$name;
?>
```

Logical Data Type Boolean

- This type of data type has 2 kind of values that is either **true** or **false.**

- The memory size of such variable is 1 byte.

- **Example:**

```
<?php
    $flag=false;
    echo $flag;
?>
```

How to Know (get) the data type of Variable:

gettype( ) function

- Syntax : gettype(variable name / value);
- Purpose : To know the data type of any variable we can use get type
- Example :

```
<?php
        $a=25000;
        $b=25.100;
        echo gettype($a);
        echo gettype($b);
?>
```

How to set the data type of Variable:

settype( ) function

Syntax : settype(variable name, data type);

Purpose: To set the data type of any variable we can use get type

Example:

```
<?php
        $a=25000;
        echo gettype($a);
        settype($a,'double');
        echo gettype($a);
?>
```

Operator & Operand

- What is Operator?
  - The symbols which are used to perform mathematical or logical operation those symbols are known as operator.
  - Like,
    - +, -, *, /, &&, | |, !=, <, >, etc.
- What is Operand?
  - Operand means the variable or value which is used before and after the operator.
  - Like
    - A+B is (Here, **A** and **B** are operand and **+** operator)

Operators in PHP

- There are different types of operators in PHP as given.
  - Arithmetic Operator
  - Relational Operator
  - Logical Operator
  - Assignment Operator

Arithmetic Operators in PHP

- Arithmetic operators are used when perform any mathematical operation.
- List of Arithmetic Operators are as given…

Sign	Uses
+	Used for Addition
-	Used for Subtraction
*	Used for Multiplication
/	Used for Division
%	Used for finding the remainder. It is also known as modules operators.

Logical Operators in PHP

- When more then two conditions are to be checked at the same time then logical operators can be helpful

Sign	Logic	Uses
&&	And	Returns TRUE If all the conditions are TRUE
||	OR	Returns TRUE If any of the condition is TURE
!	NOT	Returns TRUE If Condition is not TRUE

Relational Operators in PHP

- When variables are related to some value at that the relational operator is used.

Operator	Description
==	This operator is used to compare the two variables or the values to check the equality. It checks only the value and not the data type.

Operator	Description
===	This operator is used for comparing the equality of the variables or the values to check the equality. It will check value and data types also.
!=	This operator is for not equality. It will return true if the values are not equal. It will checks the only values not data type.
!==	This operator is also for not equality. It will compare two values and data types too.

Operator	Description
<	Used for less than
>	Used for greater than
>=	Used for greater than equal to
<=	Used for less than equal to

Assignment Operators in PHP

- There are different assignment operators as given.

Operator	Description
=	Assigning the value $a=5
*=	$a*=10 is similar to $a=$a*10
/=	$a/=10 is similar to $a=$a/10
+=	$a+=10 is similar to $a=$a+10
-=	$a-=10 is similar to $a=$a-10
%=	$a%=10 is similar to $a=$a%10
.=	$a.=$a is similar to $a=$a.$a used for concatenation of the variable

String Operators in PHP

- The operator . (dot) is used for the concatenation of the two strings and so . (dot) is also known as **String Operator.**

WRITING OUTPUT COMMAND

Syntax : void echo (string $arg1 [, string $.])

Purpose : To writing output to a web page.

Example :
```
<?php
$a="Hello";
$b=$a."World";
```

Output : **Hello World**

Conditional Statements

- Introduction:
  - While we working with PHP we can use such kind of control structure to get particular output with branching.
- If as a decision control structure:
  - if (Statement)
    - To check the given condition. If condition is TURE then it execute given statements.

IF…(SIMPLE)

Syntax :

```
if(condition)
{
        Statements when condition is true;
}
```

Purpose : It will process only if the condition is true otherwise nothing.

Example :

```
<?php
        $a=5;
        if($a>1)
        {       echo "It is Greate";
        }
?>
```

IF…ELSE

Syntax :

```
if(condition)
{
        Statements when condition is true;
}
else
{
        Statements when condition is false;
}
```

Purpose : It will process first section if the condition is true otherwise it executes the second section if the condition is false.

Example :

```
<?php
   $age=19;
   echo ("<br>Age =".$age);
   if($age>18)
         echo("<br>You can GO for VOTE");
   else
         echo("<br>You can not GO for VOTE");
?>
```

IF…ELSE IF (LADDER)

Syntax :if(condition)

{ Statements when condition is true; }

else if(condition)

{ Statements when condition is true; }

else

{ Statement when all condition is false;}

Purpose : It will process first section if first condition is true, or it will process second section if second condition is true, otherwise it executes else section if all the condition is false.

Example :

```
<?php
   $no=25;
   echo("No =".$no."<br>");
   if($no>0)
         echo "Number is Positive";
   else if($no<0)
         echo "Number is Negative";
   else
         echo "Number is ZERO";
?>
```

NESTED IF STATEMENT

Syntax :

```
if(condition)
{       if(condition)
        { When condition is true;  }
}
else
{ Statement when all condition is false;}
```

Purpose : Nested means one if inside the other if. If first condition will be true then it will check the other if, both condition will be true then it will execute given statement.

Example :

```
<?php
   $age=22;
   $gen="M";
   if($gen=="M")
   {
         if($age>21)
               echo "Gentalman You can Marry";
         else
               echo "Gentalman You cannot Marry";
   }
?>
```

Switch Case

- The switch statement causes a particular group of statements to be chosen from several available groups.
- The selection is tasted upon the current value of an expression that is included within the switch statement.
- Switch case is a built in multi-way decision statement.
- It tests value of given variable (or Text Expression) against a list of case values & block of statements associated with it is executed when a match is found.

- The general form of the switch statement is:

switch(expression)

Here expression result is an integer value or char type.

Syntax :

```
switch(TxtExpression)
{       Case TxtExp1:
                Statements;
        Case TxtExp2:
                Statements;
        Case TxtExpN:
                Statements;
        Default:
                Statements;
}
```

- Purpose : Switch case statement will provide us easy features of working it will provide selection based switching in group of statements. After completion of group execution we must put a break; statement to stop execution in each group. We can use either integer number or character for switching from list of group.

- Example :

```
<?php
switch("G")
{     case "R":
                echo "Your choice RED";
                break;
      case "G":
                echo "Your choice GREEN";
                break;
      case "B":
                echo "Your choice BLUE";
                break;
      default:
                echo "Your choice Black";
                break;
}
?>
```

- Loops
  - Loops are used when we want to execute a part for a block of statements for specified number of times or depending on some specified condition.
  - The main thing about any of the loop is that it executed for the number of times based on the logical expression (condition) that is specified to the particular loop.
  - When loop checks for the condition to execute at that time there are two types of loops available.
  - Entry Control Loop

Looping Structure

Entry Control Loop

- The loop which is checked before executing the statements of loop is called as Entry Controlled loop.
- This loop is first checked and if it is true then only the statements under that loop are executed and if it is not true then it transfers the control at to the end of the loop. They are as follow.
  - For( ) loop
  - While( ) loop

FOR... LOOP

Syntax :

```
for(initialization; condition; increment or decrement)
{
        statements;
}
```

Purpose : To provide a looping while condition will be true we can use for loop.

initialization : Initialize the value of variable.

test Condition : Test condition at here.

Increment/decrement: To increase or decrease the value.

Example :

```
<?php
   $a=1;
   for($a=1;$a<=10;$a++)
         echo "<br>".$a;
?>
```

WHILE… LOOP

Syntax :

```
while(test condition)
{
        statements;
}
```

Purpose : To provide a looping till condition will be true we can use while loop.

- While loop will check the condition first.
- If the condition will be true then it will execute the statements given in the loop, after once complete the loop, it will again and again check the condition if condition will false then it will auto exit for the loop.

Example :

```
<?php
   $a=10;
   while($a>=1)
   {
         echo "<br>".$a;
         $a=$a-1;
   }
?>
```

Looping Structure

Exit Control Loop

- The loop which is checked after executing the statements of loop is called as exit controlled loop.
- This loop is not checked in the began of loop but checks after execution of statement.
- While we want to execute any loop at least once at that time we can use this looping statement. They are as follow.
  - Do…while( )

DO…WHILE LOOP

Syntax :

```
do
{
        statements;
}while(test condition);
```

Purpose : To provide a looping till condition will be true we can use do…while loop.

- Do…while loop will check the condition after execution of statements provided in to loop.
- If the condition will be true then it will again execute the statements given in the loop, if condition will false then it will auto exit for the loop.

Example :

```
<?php
   $a=10;
   do
   {
         echo $a;
         $a=$a+1;
   }
   while($a<10);
?>
```

Nesting of Loop

- When one looping statement is within another looping statement is called as nesting of loop.
- The nesting may continue up to any desired levels.
- There are two nesting statement :
  - 1) Break Statement
  - 2) Continue Statement

Break Statements.

- Syntax : break;
- Purpose : The break command will break the loop.
- Example :

```
<?php
        for($i=1;$i<=10;$i++)
        {       if($i==5)
                        break;
                echo "<br>Num. is ".$i;
        }
?>
```

Continue Statements.

- Syntax : continue;
- Purpose : The continue command will break the current loop and continue with the next loop.
- Example :

```
<?php
        for($i=1;$i<=10;$i++)
        {
                if($i==5)
                        continue;
                echo "<br>Num. is ".$i;
        }
?>
```

PHP Arrays :

What is an Array?

- An array is a storage area where the variable can hold more then one value.
- It is like an ordered map.
- This map consists of values and keys.
- The keys are referred to as an index.
- In PHP there are two types of ARRAYS.
  - Single Dimensional Array
  - Two or Multi Dimensional Array

CONN..

Single Dimensional Array

- A single dimensional array can be defined using a function array( ).
- **Syntax :**

  $variablename=array(element1, element2...)

- **Example :**

```
$name=Array("1","Manoharsinh A. Gohil","Lathi");
echo "<br>".$name[0];
echo "<br>".$name[1];
echo "<br>".$name[2];
```

Single Dimensional Array *with Keys*

- The keys and the values declared in array should be separated by “=>”.

- **Syntax :**

```
$variablename=array(key1=>value1,
                    key2=>value2…)
```

CONN..

- **Example:**

```
<?php
  $name=Array("no"=>1,"name"=>"MAG",

       "city"=>"Lathi");
  echo "<br>".$name["no"];
  echo "<br>".$name["name"];
  echo "<br>".$name["city"];
?>
```

PHP Arrays :

Two or Multi Dimensional Array

- A multi or tow dimensional arrays consists of array inside that another array.
- This type of array can be considered as a table having rows and columns.
- A table consists of many rows and each row consists of many columns.
- **Syntax :**

```
$variable=array(array(value1, value2..),
array(value1, value2..));
```

- **TWO Dimension Array Example:**

```
<?php
   $name=Array(array("1","Ravi"),
   array("2","Raj"));
   echo "<br>".$name[0][0];
   echo "<br>".$name[0][1];
   echo "<br>".$name[1][0];
   echo "<br>".$name[1][1];
?>
```

Two or Multi Dimensional Array with KEY

- **Syntax :**

```
$variable=array(key1=>array(childKey1=>value1,
                            childKey2=>value2..),
                key2=>array(childKey1=>vlaue1,
                            childKey2=>value2..));
```

- Purpose : To define a multi dimensional array with key.

Conn..

- **Example:**

```
<?php
   $student=Array("rno1"=>array("no"=>"1","name"=>"
   Mit"),
         "rno2"=>array("no"=>"2","name"=>"Harsh"),
         "rno3"=>array("no"=>"3","name"=>"Monarch"));

   echo "<br>".$student["rno1"]["no"];
   echo "<br>".$student["rno1"]["name"];
   echo "<br>".$student["rno2"]["no"];
   echo "<br>".$student["rno2"]["name"];
   echo "<br>".$student["rno3"]["no"];
   echo "<br>".$student["rno3"]["name"];
?>
```

FOREACH

Syntax : foreach($oldArrayVariable as $newArray)

Purpose : To work with array, we can use foreach loopin statement.

foreach will read all the elements of an array one by one.

while using array the number of items are not known so instead of conting the number of items ‘foreach’ loop is used.

Foreach with Single Dimension

Example :

```
<?php
        $array=array("Banana","Mango","Graps");
        foreach($array as $num)
        {
                echo "<br>".$num;
        }
?>
```

Foreach with Multi Dimension

Example :

```
<?php
  $student=Array("rno1"=>array("no"=>"1","name"
  =>“                                    Mit"),
       "rno2"=>array("no"=>"2","name"=>"Harsh"),

       "rno3"=>array("no"=>"3","name"=>"Monarch"
  ));

  foreach($student as $data)
  {    echo "<br>".$data['no'];
```

User Defined Functions

- The library functions are those functions whose definitions is inbuilt in PHP. There is no need to write the definition for library functions.
- Some times the programmer is in need to use some codes many time. It will add the need of UDF in programming filed.
- We can write definitions for required functions which are known as **User Defined Functions.**

Advantages of UDF are as below:

- It facilitates top-down modular programming.
- The length of the source code of program is reduced.
- A function may be used at several points instead of writing the same code again and again.

Parts of UDF

- In PHP we must define a function using keyword “function”.
- The name of the function.
- The optional argument that contains the variable.
- The statements which are to be written inside the function
- The keyword return with the variable if the function returns any value.

UDF Syntax

```
function function_name(arguments)
{
  Statement…
  …..
  [return ]
}
```

No argument and no return value

- This type of function does not take any argument and does not have any return value.
- Example:

```
<?php
    function welcome()
    {
        echo “WelCome to My Page”;
    }
    welcome();
?>
```

With argument no return value

- This type of function takes some argument and does not have any return value.
- Example:

```
<?php
    function welcome($nm)
    {
        echo $nm, " WelCome to My Page";
    }
    welcome("Mr. Ravi");
?>
```

WITH ARGUMENT AND WITH RETURN VALUE

- This type of function takes some argument and return some values.
- Remember that before this type of function we must use variable of output command.
- Example:

WITH ARGUMENT AND WITH RETURN VALUE

- Example:

```
<?php
    function sum($v1,$v2)
    {       $tot=$v1+$v2;
            return $tot;
    }
    $a=sum(20,30);
    echo $a;
?>
```

Default argument value

- When programmer wants to provide some default argument to a function, at that time in PHP function with default value is used.
- When the function is called if no argument is passed then it will return the default value which is set in function definition.
- If any argument will passed then it will overwrite the value which is passed.

Default argument value

- Example :

```
<?php
  function name($str="Welcome to World")
  {
    echo $str;
  }
  name("Wel Come To EARTH");
  echo "<br>";
  name();
?>
```

VARIABLE FUNCTION

- PHP supports the variable function.
- Variable has the value of the function name and if the function is called using that variable then it will call the function having the value same as function name.

- We can define variable as function name.
  - $funname="sum";
  - $funname(a,b);

VARIABLE FUNCTION

- Example :

```
<?php
    function witharg($value)
    {           echo $value;            }
    function withoutarg()
    {           echo "Welcome";        }
    $value="witharg";
    $value("Hello World");
    $value1="withoutarg";
    $value1();
?>
```

VARIABLE SCOPE:

- The scope of variable means lifetime of variable.
- The scope of variable is use when functions are used.
- Scope of variables are as given:
  - Local Variable
  - Global Variable
  - Static Variable

Local Variable

- The variables which are used only within the function is known as ***local variable***.
- The scope of the variable is till the function ending only.

Local Variable

- Example:

```
<?php
function A()
{  $a="Monarch";
}
function B()
{  echo $a;
}
A();
B();
?>
```

- It will display an error variable is not defined…

GLOBAL VARIABLE:

- Global variables are the variable which are defined with **global** keyword.
- This kind of variable can be used in different functions too…
- The function defined with **global** keyword can be used in full of the webpage.
- To define global variable the keyword global is to be used.
  - global $a;

Global Variable

- Example:

```
<?php
    function SetName()
    {           global $a;
                $a="Raju";
    }
    function GetName()
    {           echo $GLOBALS['a']  ;
    }
    SetName();
    GetName();
?>
```

STATIC VARIABLE:

- In static variable, the value of the variable will not change even thought he page is refreshed.
- The static variable holds the value which was set and it keeps that value till the page is closed.
- The static variable declared with **static** keyword.
  - Static $a=1;

Static Variable

- Example:

```
<?php
    function increment()
    {   static $a=0;
        $a++;
        echo "<br>",$a;
    }
    increment();
    increment();
    increment();
?>
```

Variable Length Argument Function

- **func_num_args**
- **func_get_arg**
- **func_get_args**

- **func_num_args()**
  - This function will count the number of the arguments which are passed to the function when called.
  - The return type will be the integer.

FUNC_NUM_ARGS()

- Example:

```
<?php
    function a()
    {
        $argnum=func_num_args();
        echo $argnum;
    }
    a(40,50,60);
?>
```

FUNC_GET_ARG()

- To use particular argument we can use this function.
- It accepts and integer offset.
- If the offset is out of range then it will generate the warning.
- The offset starts from zero.

FUNC_GET_ARG()

- Example:

```
<?php
    function a()
    {
        $argnum=func_get_arg(2);
        echo $argnum;
    }
    a(40,50,60);
?>
```

FUNC_GET_ARGS()

- This function will display the array of the arguments which are obtained when they are passed during the call of function.

- Syntax :
  - Array func_get_args();

FUNC_GET_ARGS()

- Example:

```
<?php
    function a()
    {   $num=func_num_args();
        $array=func_get_args();
        for($i=0;$i<$num;$i++)
        {           echo $array[$i];
        }
    }
    a(40,50,60);
?>
```

- List of variable function is as given.
  - gettype()
  - settype()
  - isset()
  - unset()

How to Know the Data Type of Variable:

gettype( ) function

- Syntax : gettype(variable name / value);
- Purpose : To know the data type of any variable we can use get type
- Example :

```
<?php
        $a=25000;
        $b=25.100;
        echo gettype($a);
```

How to set the data type of Variable:

settype( ) function

Syntax : settype(variable name, data type);

Purpose: To set the data type of any variable we can use get type

Example:

```
<?php
    $a=25000;
    echo gettype($a);
    settype($a,'double');
    echo gettype($a);
?>
```

Syntax : Boolean isset(MixedVariableName);

Purpose:

To check that any value is stored in variable or not.

Example:

```
<?php
        $value="";
        if(isset($value))
                echo "Has some value";
        else
                echo "Has No Value";
?>
```

Syntax : unset(variableName);

Purpose:

To remove variable from memory we can use unset.

Example:

```
<?php
   $a=10;
   unset($a);  //This will remove
   variable $a from memory
   echo $a;    //It will display ERROR
?>
```

String Functions :

- String functions are used to perform some operations with the string.

chr	ord	strtolower
strtoupper	strtoupper	strlen
ltrim	rtrim	trim
substr	strcmp	strcasecmp
strpos	strrpos	strstr
stristr	str_replace	strrev
echo	print	

String Functions :

- Chr

Syntax : chr(ascii code in integer)

Purpose : This function is used to convert the ascii code values into character value.

Example :

```
<?php
echo chr(65);
?>
```

Output :

A

String Functions :

- ord

Syntax : ord(character code in string)

Purpose : This function is used to convert the character code values into ascii value.

Example :

```
<?php
echo ord("A");
?>
```

Output :

65

String Functions :

- strtolower

Syntax : string strtolower(string variable)

Purpose : This function returns all the characters of the string into lower case.

Example :<?php

echo strtolower("MONARCH");

?>

Output :

monarch

String Functions :

- strtoupper

Syntax : string strtoupper(string variable)

Purpose : This function returns all the characters of the string into Upper case.

Example :<?php

echo strtoupper("monarch");

?>

Output :

MONARCH

String Functions :

- ucfirst

Syntax : string ucfirst(string variable)

Purpose : This function returns first character of the string into Upper case.

Example :<?php

echo ucfirst("monarch");

?>

Output :

Monarch

String Functions :

- ucwords

Syntax : string ucwords(string variable)

Purpose : This function returns first character of all the words into Upper case.

Example :<?php

echo ucwords("monarch computer");

?>

Output :

Monarch Computer

String Functions :

- strlen

Syntax : Integer strlen(string variable)

Purpose : This function returns total number of characters with space present in the string.

Example :<?php

echo strlen("monarch computer");

?>

Output :

String Functions :

- ltrim

Syntax : string ltrim(string variable [,String Char])

Purpose : This function also known as left trim. This function is used to remove the blank space or specified character from beginning (left side) of the string..

Example :<?php

```
echo strlen(ltrim("    monarch"));
echo ltrim("***monarch***","*");
?>
```

String Functions :

- rtrim

Syntax : string rtrim(string variable[,String Char])

Purpose : This function also known as right trim. This function is used to remove the blank space or specified character from ending (right side) of the string..

Example :<?php

```
echo strlen(rtrim("    monarch"));
echo rtrim("***monarch***","*");
?>
```

String Functions :

- trim

Syntax : string trim(string variable)

Purpose : This function is used to remove the blank space or specified character from beginning and ending (left & right side) of the string..

Example :<?php

```
echo strlen(trim("    monarch
"));

echo trim("***monarch***","*");
?>
```

String Functions :

- substr

Syntax: string substr(string variable, integer position, integer length)

Purpose : This function is used to retrieved specified part of the string.

Location index starts from 0.

Example:<?php

echo substr("Computer",0,3);

?>

Output : Com

String Functions :

- strcmp

Syntax: Integer strcmp(String1, String2)

Purpose : This function is used to compare two string with each other.

Note : It is case sensitive…

If string1 is greater than string2 then it will return>0.

If string2 is greater then string1 then it will return<0.

If the comparison of both string are equal

STRING FUNCTIONS :

- strcasecmp

Syntax: Int strcasecmp(String1, String2)

Purpose : This function is used to compare two string with each other.

Note : It is NOT case sensitive…

If string1 is greater than string2 then it will return>0.

If string2 is greater then string1 then it will return<0.

If the comparison of both string are equal

String Functions :

- strncasecmp

Syntax : Int strncasecmp(String1, String2,int length)

Purpose **:** This function is used to compare two string with each other using fixed length given.

Note : It is NOT case sensitive…

If string1 is greater than string2 then it will return>0.

If string2 is greater then string1 then it will return<0.

String Functions :

- strpos

Syntax : Int strpos(String, Character)

Purpose **:** This function is used to find the position of **first** occurrence of the character in the string.

Note : This function is case sensitive.

Example:

```
echo strpos("NITIN","I");
```

Output :

1

String Functions :

- strrpos

Syntax : Int strrpos(String, Character)

Purpose **:** This function is used to find the position of **Last** occurrence of the character in the string.

Note : This function is case sensitive.

Example:

```
echo strrpos("NITIN","I");
```

Output :

3

String Functions :

- strstr

Syntax : String strstr(String variable, character)

Purpose : This function is used to get string from specified character to end of string.

Note : This function is case sensitive.

Example:

echo strstr("NITIN","I");

Output :

ITIN

String Functions :

- stristr

Syntax : String stristr(String variable, character)

Purpose : This function is used to get string from specified character to end of string.

Note : This function is NOT case sensitive.

Example:

```
echo stristr("NITIN","i");
```

Output :

ITIN

String Functions :

- strrev

Syntax : String strrev(String variable)

Purpose **:** This function is used to get string in reverse order.

Example:

```
echo strrev("LATHI");
```

Output :

IHTAL

String Functions :

- str_replace

Syntax : String str_replace(Search String/Array, Character to Replace, Original String)

Purpose **:** This function is used to find specified character and replace it with given character.

Note : str_replace function is case sensitive.

Example:

echo str_replace("a","*","Monarch");

String Functions :

- str_ireplace

Syntax : String str_ireplace(Search String/Array, Character to Replace, Original String)

Purpose **:** This function is used to find specified character and replace it with given character.

Note : str_ireplace function is NOT case sensitive.

Example:

String Functions :

- strrchr

Syntax : strrchr(String, Character to be search)

Purpose : This function is used to retrieve the string from last occurrence of given character.

Note : strrchr function is case sensitive.

Example:

```
echo strrchr("Nana Nani","N");
```

Output :

Nani

String Functions :

- strval

Syntax : String strval(Variable of other data type)

Purpose : This function is used to convert any data type value of PHP into string value and will return string value.

Example:

```
<?php
        $digit=12345;
        echo strval($digit);
?>
```

String Functions :

- echo

Syntax : echo “String Value”, variable list

Purpose : This function is used display any value on the web-page. This function will not return anything and so that return type of this function is void.

Example:

```
<?php
        $digit=12345;
        echo “Value Given :”,$digit;
?>
```

String Functions :

- print

Syntax : print(argument)

Purpose : This function is used display any value on the web-page. This function will not support multi variables in same statements.

Example:

```
<?php
   $a="WelCome";
   $b=" Bye";
   print $a;
   print $b;
?>
```

String Functions :

- addslashes

Syntax : string addslashes(String variable)

Purpose : This function is used to insert the backward slash (\) in front of backslash (\), double quote("), single quote('), and NULL.

Example:

```
<?php
   echo "monarch's";
   echo addslashes("monarch's");
?>
```

Output :

monarch's
monarch\'s

String Functions :

- stripslashes

Syntax : string stripslashes(String variable)

Purpose **:** This function is used to remove the backward slash (\) from in front of backslash(\), double quote("), single quote('), and NULL.

Example:

```
<?php
   echo "monarch\'s";
   echo stripslashes("monarch\'s");
?>
```

Output :

monarch \'s

monarch's

MATH FUNCTIONS :

- abs( )

Syntax: mixed abs(mixed number)

Purpose : This function is used to find the absolute value of given number. If value is in integer then the return type is integer and if it is float then the return type is float.

Example: echo abs(-65);

echo abs(-42.5);

Output : 65

42.5

Math Functions :

- ceil( )

Syntax: int ceil(float value)

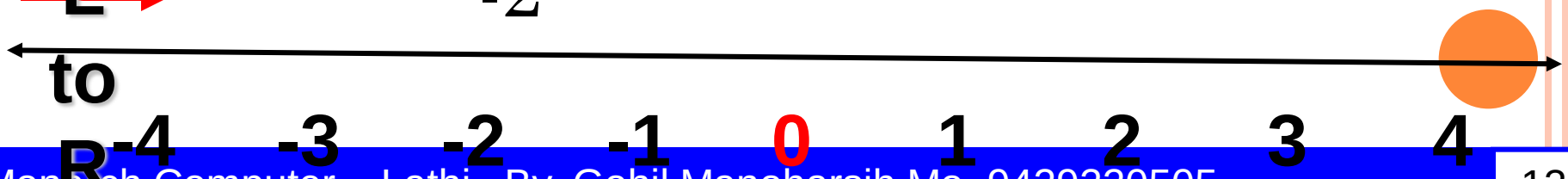
Purpose : This function will return nearest integer larger value for given float value.

Example: echo ceil(5.7);

echo ceil(-2.2);

Output : 6

-2



Math Functions :

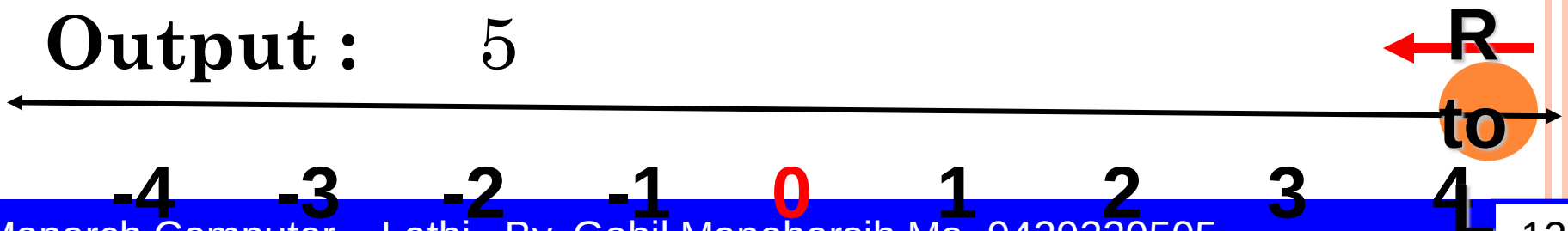
- floor( )

Syntax: int floor(float value)

Purpose : This function will return nearest integer small value for given float value.

Example: echo floor(5.7);

Output : 5



MATH FUNCTIONS :

- round( )

Syntax: float round(float value, int precision)

Purpose : This function will round the floating value as given precision.

Example: echo round(5.7565,2);
echo round(2.2222,3);

Output : 5.76
2.222

Math Functions :

- fmod( )

Syntax: mixed fmod(value1, value2)

Purpose **:** This function will act as an operator module which is used to find reminder between two digits.

Example:

```
echo fmod(5.75,2.25);
echo "<br>";
echo fmod(4.5,1.5);
```

Output :

```
1.25
0
```

Math Functions :

- min( )

Syntax: mixed min(array or value list)

Purpose : This function will return the minimum value from given list.

Example:

```
echo min(3,55,1,9);
echo "<br>";
$vallist=array(44,55,33,99,22);
echo min($vallist);
```

Output :

```
1
22
```

Math Functions :

- max( )

Syntax: mixed max(array or value list)

Purpose : This function will return the maximum value from given list.

Example:

```
echo max(3,55,1,9);
echo "<br>";
$vallist=array(44,55,33,99,22);
echo max($vallist);
```

Output :

```
55
99
```

Math Functions :

- pow( )

Syntax: mixed pow(base value, exponential value)

Purpose : This function is used to find the power of base value.

Example: echo pow(3,2);

Output : 9

Math Functions :

- sqrt( )

Syntax: mixed sqrt(mixed value)

Purpose **:** This function is used to find square root of the given number.

Example: echo sqrt(16);

Output : 4

Math Functions :

- rand( )

Syntax: int rand(int min, int max)

Purpose : This function will generates the random integer value each time the page is refreshed.

Example: echo rand(1,100);

Output : Each time different…

Math Functions :

- bindec( )

Syntax: Mixed bindec(string binary value)

Purpose **:** This function converts the number from binary to decimal format.

Example: echo bindec(“110110”);

Output : 54

Math Functions :

- octdec( )

Syntax: Mixed octdec(string octal value)

Purpose **:** This function converts the number from octal to decimal format.

Example: echo octdec("54");

Output : 54

Math Functions :

- hexdec( )

Syntax: Mixed hexdec(string hexadecimal value)

Purpose : This function converts the number from hexadecimal to decimal format.

Example: echo hexdec("A7");

Output : 167

Math Functions :

- decbin( )

Syntax: Mixed decbin(string decimal value)

Purpose **:** This function converts the number from decimal to binary format.

Example: echo decbin("54");

Output **:** 110110

Math Functions :

- decoct( )

Syntax: Mixed decoct(string decimal value)

Purpose **:** This function converts the number from decimal to octal format.

Example: echo decoct(“44”);

Output **:** 54

MATH FUNCTIONS :

- dechex( )

Syntax: Mixed dechex(string decimal value)

Purpose **:** This function converts the number from decimal to hexadecimal format.

Example: echo dechex("167");

Output **:** a7

Date Function

- To work with date we can use date function

Date
getdate
setdate
checkdate
time
mktime

DATE

Syntax :

date date(‘Format’);

Purpose :

To return date with specified format we can use date function.

Example :

```
<?php
        echo date('d-F-Y');
?>
```

Output :

28-December-2013

a	am or pm (lowercase)
A	AM or PM (uppercase)
d	Day of month (number with leading zeroes)
D	Day of week (three letters)
e	Timezone identifier
F	Month name
h	Hour (12-hour formatleading zeroes)
H	Hour (24-hour formatleading zeroes)
g	Hour (12-hour formatno leading zeroes)
G	Hour (24-hour formatno leading zeroes)
i	Minutes
j	Day of the month (no leading zeroes)
l	Day of the week (name)

L	Leap year (1 for yes, 0 for no)
m	Month of year (numberleading zeroes)
M	Month of year (three letters)
n	Month of year (numberno leading zeroes)
s	Seconds of hour
S	Ordinal suffix for the day of the month
r	Full date standardized to RFC 822 (http://www.faqs.org/rfcs/rfc822.html)
U	Time stamp
y	Year (two digits)
Y	Year (four digits)
z	Day of year (0-365)
Z	Offset in seconds from GMT

GETDATE()

Syntax :

Array getdate('Format');

Purpose :

To return date in different format and store it as an array we can use getdate function.

Example :

```
<?php
    $dt=getdate(seconds);
    foreach($dt as $key=>$val)
        echo "$key = $val<br>";
?>
```

Output :

seconds = 4
minutes = 38
hours = 19
mday = 28
wday = 6
mon = 12
year = 2013
yday = 361
weekday = Saturday
month = December

SETDATE()

Syntax :

Void setdate(int year, int mon, int day);

Purpose :

To set user defined date we can use setdate function.

Example :

```
<?php
    $date = new DateTime();
    $date->setDate(1979,06,11);
    echo $date->format('D:d-F-Y');
?> Output : Mon : 11-June-1979
```

Syntax :

Boolean checkdate(int month, int day, int year)

Purpose :

This function will check whether the given date is in proper format then TRUE or not then FALSE.

Example :

```
<?php
$format=checkdate(06,11,1979);
if($format==true)
    echo "Given Date is Valid";
else
    echo "Given Date is NOT Valid";
?> Output : Given Date is Valid
```

Syntax :

DateTime time()

Purpose :

This function will returns the current time from January 1, 1970 to current time in seconds format.

Example :

```
<?php
    echo time();
?>
```

Output : 1388261763

Syntax :

Date mktime(int hour, int min, int sec, int month, int date, int year)

Purpose :

This function will returns time stamp that can be used with date or getdate function.

Example :

```
<?php
    $userdate=mktime(1,50,20,06,11,1979);
    echo date('d-m-Y',$userdate);
?>
```

Output : 11-06-1979

Array Handling Functions :

- count( )

Syntax :

int count(Array variable)

Purpose :

This function is used to count number of element of array.

Example :

```
<?php
    $student=array(1,"Ravi","Lathi");
    echo "No of Ele:".count($student);
?>
```

OutPut :

Element Are :3

ARRAY HANDLING FUNCTIONS :

- list( )

Syntax :

void list(variable1, variable2…)=ArrayVar

Purpose:

To assign value of array in individual variable list.

Example :

```
$student=array(1,"Ravi","Lathi")
list($no,$name,$city)=$studen
echo "<br>No = " $no;
```

OutPut :
No = 1
Name = Ravi
City = Lathi

Array Handling Functions :

- is_array( )

Syntax :

int is_array(variable name)

Purpose:

This function is used to check whether the given variable is an array variable or not.

Example:

```
$student=array(1,"Ravi","Lathi");
if(is_array($student))
        echo "It is array variable";
else
        echo "Sorry its not an array variable";
```

OutPut :

It is array variable

Array Handling Functions :

- in_array( )

Syntax :

Boolean in_array(val. to search, array vari.)

Purpose :

This function is used to search the given value is present in array or not. If value is found then it will return logical true else false.

Example:

```
$student=array(1,"Ravi","Lathi");
if(in_array("Ravi",$student))
    echo "Value is found";
else
    echo "Value is NOT Found";
```

OutPut :

Value is found

Array Handling Functions :

- current( )

Syntax :

mixed current(array variable)

Purpose :

This function will return the current item element from selected array variable basically new declared array will return current records as first record.

Example:

```
$student=array("Ravi","Umang");
echo current($student);
```

OutPut :
Ravi

Array Handling Functions :

- next( )

Syntax :

mixed next(array variable)

Purpose :

This function will moves the pointer to the next position in array and returns the value of element.

Example:

```
$student=array("Ravi","Umang");
echo current($student);
echo "<br>";
echo next($student);
```

OutPut :
Ravi
Umang

Array Handling Functions :

- prev( )

Syntax :

mixed prev(array variable)

Purpose :

This function will moves the pointer to the previous position in array and returns the value of element.

Example :

```
$student=array("Ravi","Umang");
echo next($student);
echo “<br>”;
echo prev($student);
```

OutPut :

Umang

Ravi

Array Handling Functions :

- reset( )

Syntax :

mixed reset(array variable)

Purpose :

This function will moves the pointer to the **first** position in array and returns the value of first element.

Example:

```
$student=array("Ravi",“Umang");
echo next($student);
echo reset($student);
```

OutPut :
Umang
Ravi

Array Handling Functions :

- end( )

Syntax :

mixed end(array variable)

Purpose :

This function will moves the pointer to the **last** position in array and returns the value of last element.

Example:

```
$student=array("Ravi",“Umang”,“Uma”);
echo end($student);
echo reset($student);
```

OutPut :
Uma
Ravi

Array Handling Functions :

- each( )

Syntax :

Array each(array variable)

Purpose :

This function returns the current key and value pair from an array and advance the array cursor.

Example:

```
<?php
        $student=array(1,"Monarch");
        $each=each($student);
        print_r($each);
?>
```

Output : Array ([1] => 1 [value] => 1 [0] => 0 [key] => 0)

Array Handling Functions :

- sort( )

Syntax :

void sort(array variable)

Purpose :

This function will sort the items in the array in an ascending order.

It will reorder the index too.

Example:

```
$rno=array(2,5,4,3,1);
sort($rno);
foreach($rno as $roll)
    echo $roll;
```

OutPut :
12345

Array Handling Functions :

- rsort( )

Syntax :

void rsort(array variable)

Purpose :

This function will sort the items in the array in a descending order.

It will reorder the index too.

Example:

```
$rno=array(2,5,4,3,1);
rsort($rno);
foreach($rno as $roll)
    echo $roll;
```

OutPut :

54321

Array Handling Functions :

- asort( )

Syntax :

void asort(array variable)

Purpose :

Associative sorting function will sort the items in the array in an ascending order.

It will NOT reorder the index.

Example:

```
$rno=array(2,5,4,3,1);
asort($rno);
foreach($rno as $roll)
    echo $roll;
```

OutPut :

12345

Array Handling Functions :

- array_merge( )

Syntax :

void array_merge(array1, array2…)

Purpose :

This function is used to merge more then one array variable in array variable.

Example:

```
$no1=array(1,2,3);
$no2=array(4,5,6);
$no=array_merge($no1,$no2);
foreach($no as $roll)
    echo $roll;
```

OutPut :

123456

Array Handling Functions :

- array_values( )

Syntax :

Array array_values(array)

Purpose :

This function is used to store only values to array variable it will not return any keys/index title.

Example:

```
$student=array("Rno"=>1,"nm"=>"Ramesh");
$onlyval=array_values($student);
foreach($onlyval as $val)
    echo "<br>".$val;
```

OutPut :

1
Ramesh

Array Handling Functions :

- array_keys( )

Syntax :

Array array_keys(array)

Purpose :

This function is used to store only keys/index to array variable it will not return any values.

Example:

```
$student=array("Rno"=>1,"nm"=>"Ram");
$onlykey=array_keys($student);
foreach($onlykey as $key)
    echo $key;
```

OutPut :

Rno

nm

Array Handling Functions :

- print_r( )

Syntax :

void print_r(array vari)

Purpose :

This function is used to display the array in the format as it is declared in the code.

Example :

$student=array("Rno"=>1,"nm"=>"Ram");

print_r($student);

OutPut :

Array ([Rno] => 1 [nm] => Ram)

Array Handling Functions :

- array_key_exists( )

Syntax :

Boolean array_key_exists(value to search,
array variable)

Purpose :

This function is used to check that given string/value key is present in the array or not.

Example:

```
$student=array("Rno"=>1,"nm"=>"Ram");
if(array_key_exists("Rno",$student))
    echo "Given Key is found in array";
```

Array Handling Functions :

- array_reverse( )

Syntax :

array array_reverse(array variable)

Purpose :

To reverse the items present in the array.

Example:

```
$sub=array("PHP", "Oracle", "VB.Net");
$revsub=array_reverse($sub);
echo print_r($revsub);
```

OutPut :

```
Array ( [0] => VB.Net [1] => Oracle [2] => PHP )
```

Array Handling Functions :

- array_push( )

Syntax :

array array_push(array variable,value1,value2…)

Purpose :

To insert the item in array at end of existing array.

Example:

```
$sub=array("PHP", "Oracle");
array_push($sub,"VB.Net");
echo print_r($sub);
```

OutPut :

Array ([0] => PHP [1] => Oracle [2] => VB.Net)

Array Handling Functions :

- array_pop( )

Syntax :

array array_pop(array variable)

Purpose :

To remove the last item from array at end of existing array.

Example:

```
$sub=array("PHP", "Oracle");
array_pop($sub);
echo print_r($sub);
```

OutPut :

Array ([0] => PHP)

Miscellaneous Functions :

- Include
  - **Syntax :** include(StringFilename)
  - **Purpose :** To access contents and function from included file we can use include function.

Miscellaneous Functions : (Cont.)

- **Example :**

```
<?php
        $name="Monarch";
?>      //Save this file with nm.php name


<?php
        include(nm.php);
        echo $name;
?>      We can also use variable from called file...
```

Miscellaneous Functions : (Cont.)

- require()

Syntax : void require(string file name)

Purpose: To access various values from file provided in require function. ***It is as same as include function but it will generate fatal error if file not found.***

Example: require("myfun.php");

- Header
  - **Syntax** : header("Location:filename");
  - **Purpose** : To transfer the page from one

Miscellaneous Functions : (Cont.)

- **Example :**

```
<?php
    $id="monarch";
    $pass=“bca";
    if($id=="monarch" and $pass==“bca")
            header("location:welcome.php");
    else
            echo "Wrong ID or Password";
?>
```

Miscellaneous Functions : (Cont.)

- define

Syntax : define("variable name","value",
Boolean value for casesensitivity);

Purpose: To define variable as constant with value. The value defined with **define** will not change during the program.

Example:

```
<?php
    define("PI",3.14);
    echo PI;
?>
```

Miscellaneous Functions : (Cont.)

- constant

Syntax : constant(String Constant Var. Name);

Purpose: To store constant name as constant in variable.

Example:

```
<?php
        define("PI",3.14);
        $val="PI";
        echo $val;
        $val=constant("PI");
        echo $val;
?>
```

Miscellaneous Functions : (Cont.)

- **die**

Syntax : void die( );

Purpose: To exit from current script coding.

Example:
```
<?php
        Echo "Welcome";
        die( );
        echo "Use Of DIE Function";
?>
```

To close WebPage :
```
<script language="javascript">
        window.close( );
</script>
```

Note : It will work only with some browsers...

Miscellaneous Functions : (Cont.)

- isset

Syntax : Boolean isset(Mixed Variable Name);

Purpose: To check that any value is stored in variable or not.

Example:

```
<?php
    $value="";
    if(isset($value))
        echo "Has some value";
    else
        echo "Has No Value";
?>
```

FOPEN

- Syntax :

  handler fopen(string filename, string mode)

- Purpose :
  - This function is used to open the file or the URL.
- Example :

```
<?php
  fopen("testing.txt","r");
  fopen("d:\\monarch.txt","w");
```

FWRITE

- Syntax :

  int fwrite(file handler variable, String value)

- Purpose :
  - This function is used to writes contents into specified file.
- Example :

```
<?php
  $fp=fopen("d:\\monarch.txt","w");
  fwrite($fp,"Wel Come to MY File");
```

FREAD

- Syntax :

  int fread(file handler variable, int length)

- Purpose :
  - This function is used to read contents from specified file.
- Example :

```
<?php
  $fp=fopen("d:\\monarch.txt","r");
  $fileData=fread($fp,filesize("d:\\monarch.t
  xt"));
  echo $fileData;
?>
```

FCLOSE

- Syntax :

  fclose(file handler variable);

- Purpose :
  - This function is used to close a specified file.
- Example :

```
<?php
  $fp=fopen("d:\\monarch.txt","r");
  $fileData=fread($fp,filesize("d:\\monarch.t
  xt"));
  echo $fileData;
  fclose($fp);
```

FILE_EXISTS

- Syntax :

  boolean file_exists(string file name);

- Purpose :
  - This function is used to check that given file is exists or not. If exists then it return the logical TRUE else it will returns logical FALSE.

FILE_EXISTS

- Example :

```
<?php
  if(file_exists("d:\\monarch.txt"))
        echo "monarch.txt file is EXISTS";
  else
        echo "monarch.txt file not FOUND";
?>
```

IS_READABLE

- Syntax :

  boolean is_readable(string file name);

- Purpose :
  - This function is used to check that given file is readable or not. If readable then it return the logical TRUE else it will returns logical FALSE.

IS_READABLE

- Example :

```
<?php
  if(is_readable("d:\monarch.txt"))
        echo "monarch.txt file is
READABLE";
  else
        echo "monarch.txt file not
READABLE";
?>
```

IS_WRITEABLE

- Syntax :

  boolean is_writeable(string file name);

- Purpose :
  - This function is used to check that given file is writeable or not. If writeable then it return the logical TRUE else it will returns logical FALSE.
  - If file is read only then it will become not writable and it returns logically false.

IS_WRITABLE

- Example :

```
<?php
    if(is_writeable("d:\monarch.txt"))
            echo "monarch.txt file is
  WRITEABLE";
    else
            echo "monarch.txt file not
  WRITEABLE";
?>
```

FGETS

- Syntax :

  string fgets(file handler variable, integer length)
- Purpose :
  - This function is used to reads the contents of the file line by line.
  - The type of return is string.
  - The difference between fgets and fread is that,
    - The function fgets() reads the contents till the new line is found.
    - The function fread() reads the contents word by word.
    - To get the end of file the function feof() is to be used.

FGETS()

- Example :

```
<?php
   $fp=fopen ("d:\\monarch.txt","r");
   while(!feof($fp))
   {
     $data=fgets($fp);
     echo $data;
   }
?>
```

FGETC

- Syntax :

  string fgetc(file handler variable)
- Purpose :
  - This function is used to reads the contents of the file character by character.
  - The type of return is string.
  - The difference between fgets and fread is that,
    - The function fgets() reads the contents till the new line is found.
    - The function fread() reads the contents word by word.
    - To get the end of file the function feof() is to be used.

FGETC()

- Example :

```
<?php
   $fp=fopen ("d:\\monarch.txt","r");
   while(false !==($data=fgetc($fp)))
   {
            echo "<br>",$data;
   }
?>
```

FILE

- Syntax :

  Array file(string filename)

- Purpose :
  - This function is used to read the entire file in the array format and so the return type of this function is Array.
  - Each line in the file is stored as the new item in the array.
  - There is no requirement to open the file, it gets automatically when the file function is used.

FILE( )

- Example :

```
<?php
  $file=file("d:\\monarch.txt");
  foreach($file as $newarr)
  {
    echo $newarr;
  }
?>
```

FILE_GET_CONTENTS

- Syntax :

  string file_get_contents(string filename)

- Purpose :
  - It read the contents of the file and stores it in the string format.
  - There is no requirement to open the file, it gets automatically opened,
  - There is no need to give the length.
  - It will read till the end of the file.

FILE_GET_CONTENTS

- Example :

```
<?php
  $data=file_get_contents("d:\\monarch.txt")
  ;
  echo $data;
?>
```

FILE_PUT_CONTENTS()

- Syntax :

  void file_put_contents(string filename,

  string data to be written)

- Purpose :
  - This function writes the contents into the file.
  - There is no requirement to open the file it gets automatically opened for writing.

FILE_PUT_CONTENTS()

- Example :

```
<?php
  $data=file_get_contents("E:\\monarch.txt");
  echo $data;

  $newdata="Other Branch : Monarch - Babra";
  file_put_contents("E:\\monarch.txt",$data);

  $data=file_get_contents("E:\\monarch.txt");
  echo $data;
?>
```

FTELL

- Syntax :

  int ftell(file handler varibale);

- Purpose :
  - This function used to give the current position of the pointer in the file.
  - It returns integer value.
  - Its integer value starts from 0.

FTELL( )

- Example :

```
<?php
  $fp=fopen("d:\\monarch.txt","r");
  $data=fgets($fp,1);
  echo ftell($fp);
?>
```

REWIND

- Syntax :

  Boolean rewind(file handler variable);

- Purpose :
  - This function moves the file pointer to the start position of the line.
  - It returns true if start position is found, otherwise will returns false.

REWIND()

- Example :

```
<?php
  $fp=fopen("d:\\monarch.txt","r");
  $data=fgets($fp,10);
  rewind($fp);
  echo ftell($fp);
?>
```

COPY

- Syntax :

  Boolean copy(string oldfilename, stringnewfilename);

- Purpose :
  - This function used to copy the file with same contents into the new file.
  - It returns true on successful copy of the file else it will return false.

COPY

- Example :

```
<?php
  $ans=copy("d:\\monarch.txt","d:\\abc.txt")
  ;
  echo "File is Copied";
?>
```

UNLINK

- Syntax :

  Boolean unlink(string filename);

- Purpose :
  - This function used to delete the given file.

UNLINK

- Example :

```
<?php
  unlink("d:\\abc.txt");
?>
```

RENAME

- Syntax :

  Boolean rename(Oldfilename, NewFilename);

- Purpose :
  - This function used to rename file.
- Example :

```
<?php
  rename("d:\\monarch.txt","d:\\mon.txt");
  echo "File is Renamed";
?>
```

FILESIZE

- Syntax :

integer filesize(string filename);

- Purpose :
  - This function used to return the number of characters from file.
  - This function will work as length function of string.
- Example :

```
<?php
  echo filesize("d:\\mon.txt");
?>
```

FILEATIME

- Syntax :

integer fileatime(string filename);

- Purpose :
  - This function used to return the time of last time accessing of the file.
- Example :

```
<?php
  echo fileatime("d:\\mon.txt");
?>
```

FILEMTIME

- Syntax :

integer filemtime(string filename);

- Purpose :
  - This function used to return the time of last time modification of the file.
- Example :

```
<?php
  echo filemtime("d:\\mon.txt");
?>
```

FSEEK

- Syntax : integer fseek(file handler variable, integer offset);
- Purpose:
  - The function enables you to change your current position within a file.

FSEEK

- Example :

```
<?php
  $filename="d:\mon.txt";
  $fp=fopen($filename,"r");
  fseek($fp,27);
  $seek=fread($fp,filesize($filename));
  echo $seek;
?>
```

MOVE_UPLOADED_FILE()

- Syntax : Boolean move_uploaded_file (source file, destination location)
- Purpose:
  - The function will move and uploaded file into the new location. It returns true if the file is moved else will return false.
- Example:

```
move_uploaded_file("D:\Student.txt", "E:\");
echo "File Moved Successfully";
```