PHP Programming
Part 1

Dr. Ashok Kumar Jetawat (Chairman)

Ph.D., M.Tech, MBA, MCA, MAJM, MSW
SCJP, MCP, IGCBP, Geneva (Switzerland)

IICE College

email : drashokjain61@gmail.com
Web: http://ashokjetawat.com
(M) 9001556010

Lesson 1

PHP

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

What is PHP?

- PHP is an acronym for "PHP: Hypertext Preprocessor"
- PHP is a widely-used, open source scripting language
- PHP scripts are executed on the server
- PHP is free to download and use

To start using PHP

- Install Web Server. Ex- xampp, wamp etc.
- Install Editor. Ex- notepad ++.

PHP Syntax

A PHP script starts with

<?php

// PHP statements

?>

Comments

A comment in PHP code is a line that is not executed as part of the program.

```
// This is a single-line comment

# This is also a single-line comment
```

```
/*
This is a multiple-lines comment block
that spans over multiple lines
*/
```

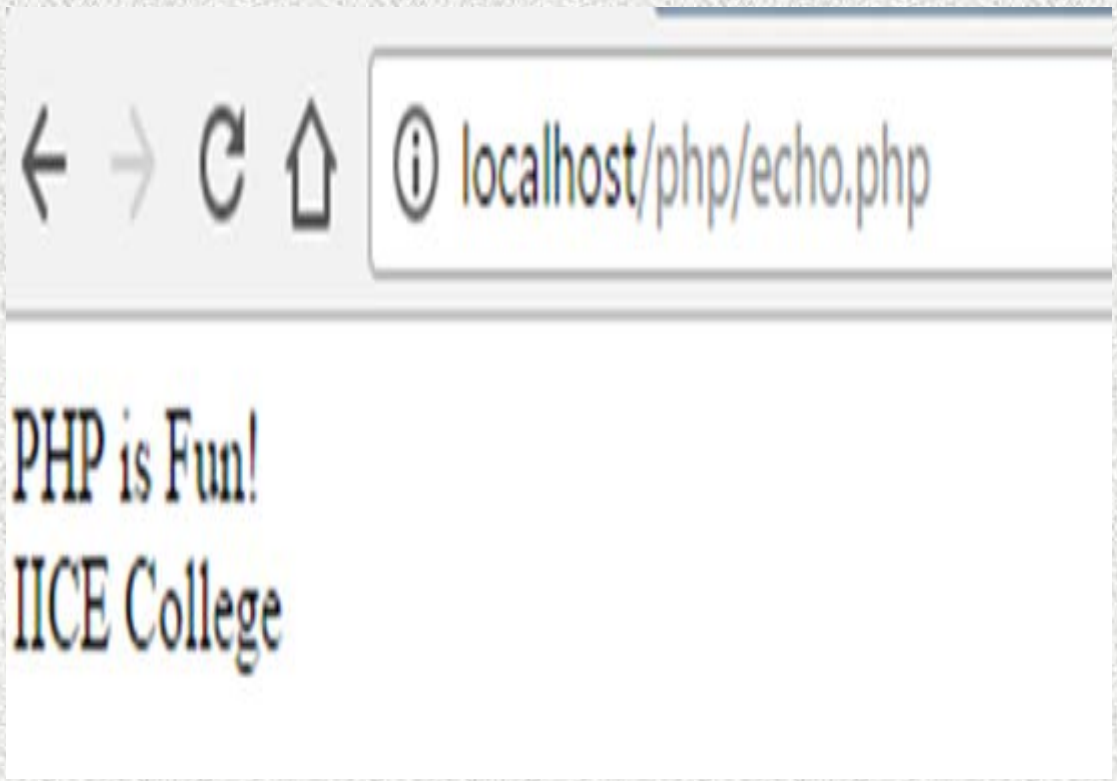
echo and print Statements

They are both used to output data to the screen.

- **echo** can take multiple parameters.
- **print** can take one argument. echo is faster than print.

Example

```
<?php

echo "PHP is Fun!</br>";

print "IICE College";

?>
```



Exercise

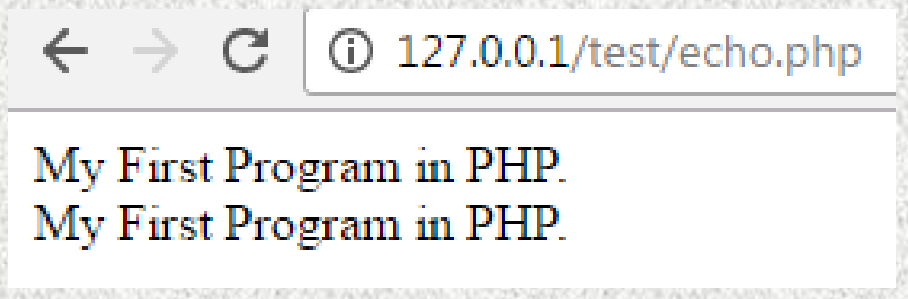
1. Write a program to print :

   " My First Program in PHP".

2. Write a program to show implementation of single and multiline comment.

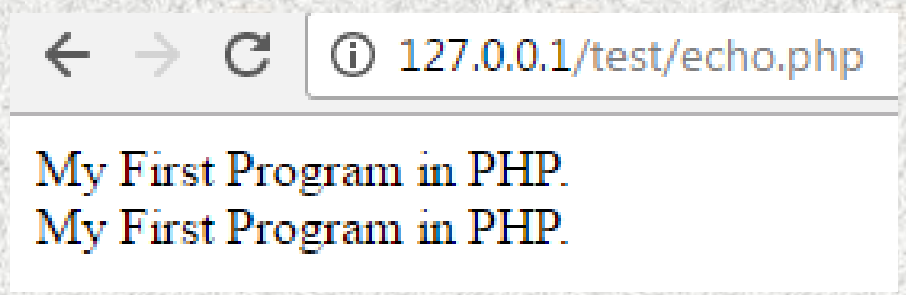
1. Write a program to print :
My First Program in PHP.

```
<?php
echo"My First Program in PHP.<br>";
Print"My First Program in PHP.";
?>
```



2. Write a program to show implementation of single and multiline comment.

```
<?php
//Use echo
#Use echo
echo"My First Program in PHP.<br>";
/*Use Print
to show content*/
Print"My First Program in PHP.";
?>
```



Lesson 2

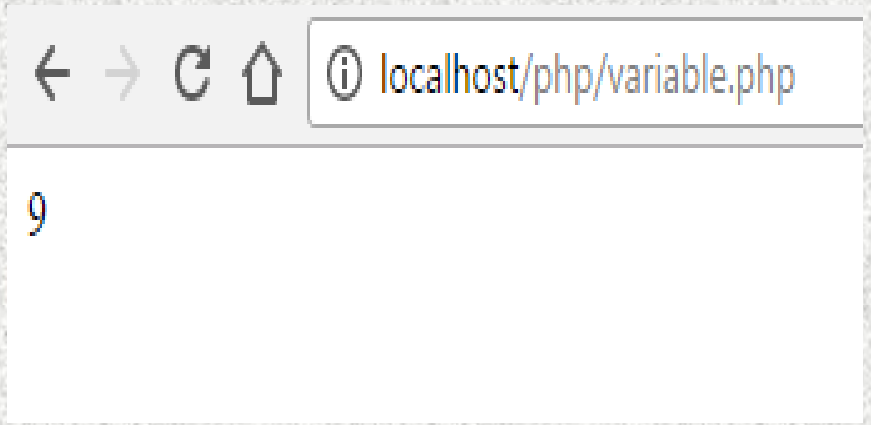
Variables

- 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 ($age and $AGE are two different variables)

Note:- Variable names are case-sensitive!

Example

```
<?php
$x=5;
$y=4;
echo $x+$y;
?>
```



PHP Variables Scope

PHP has three different variable scopes: local, global, static

```
<?php
$x = 5; // global scope
function myTest () {
$y=6; // local scope
static $z=7; // static scope
        echo "x=". $GLOBALS['x']."<br>";  //print global variable
        global $x;
   echo "x= $x </br>"; //print global variable
        echo "y= $y </br>";//print local variable
        echo "z= $z </br>";//print static variable
        }
myTest();
?>
```

Variables Scope

PHP has three different variable scopes: local, global, static

```
<?php
$x = 5; // global scope
function myTest ()
{
$y=6; // local scope
static $z=7; // static scope
echo "x=". $GLOBALS['x']."<br>";
//print global variable
global $x;
echo "x= $x </br>"; //print global variable
echo "y= $y </br>";//print local variable
echo "z= $z </br>";//print static variable
}
myTest();
?>
```

```
x=5
x= 5
y= 6
z= 7
```

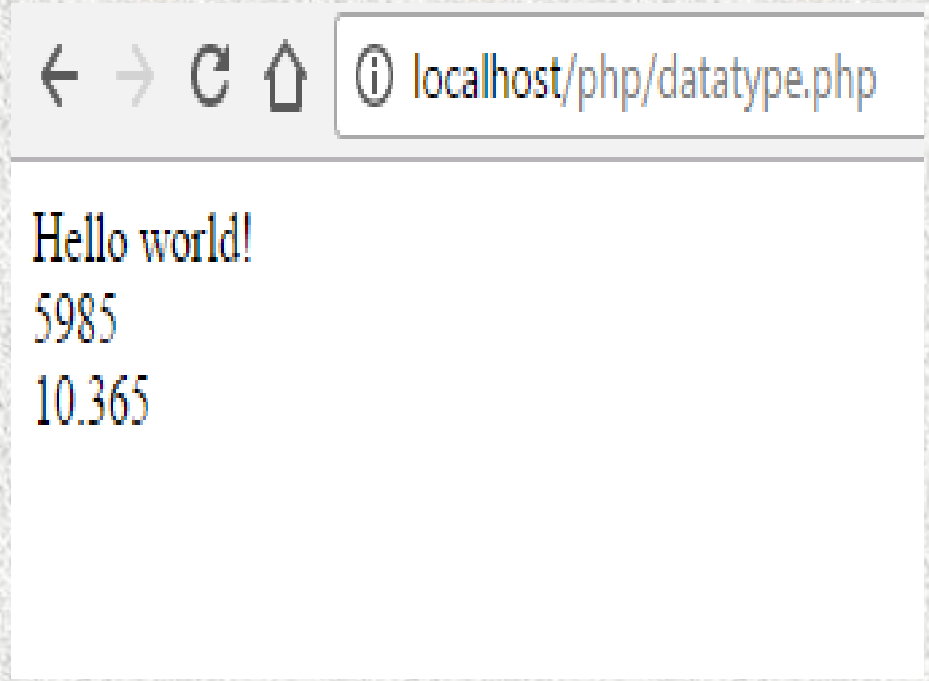
Data Types

PHP supports the following data types:

- String
- Integer
- Float / double
- Boolean
- Array

Example

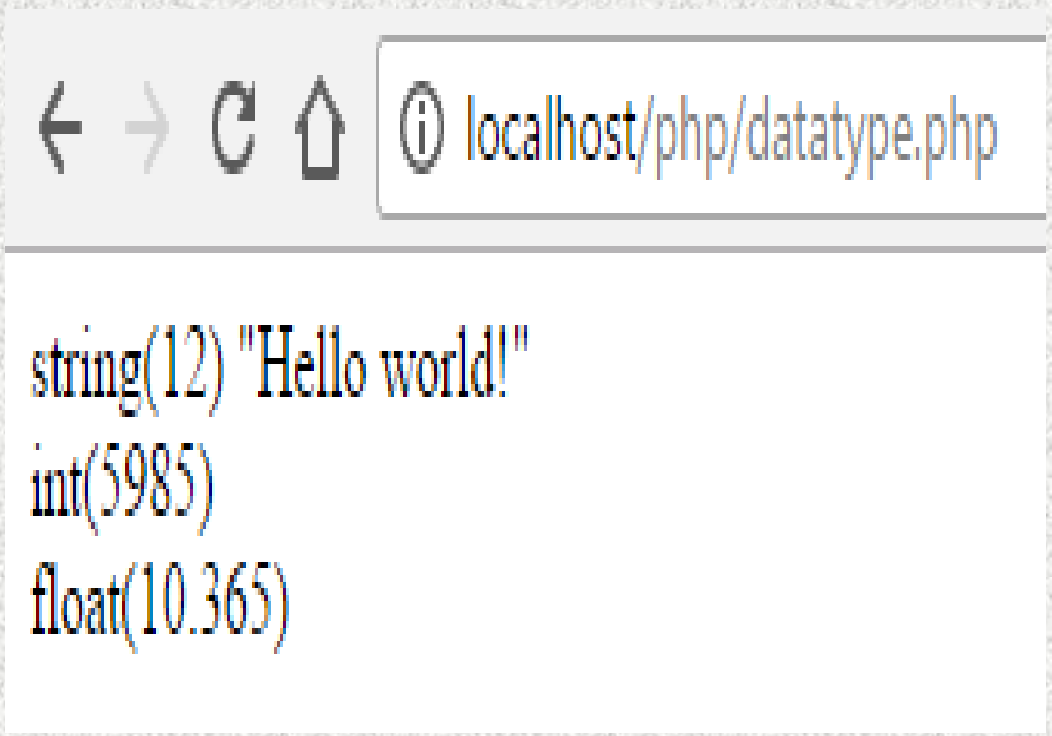
```
<?php
$a = "Hello world!";
$b = 5985;
$c = 10.365;
echo "$a</br>";
 echo "$b</br>";
  echo "$c</br>";
?>
```



Var_dump() Function

- The PHP var_dump() function returns the data type and value:

```
<?php
$a = "Hello world!";
$b = 5985;
$c = 10.365;
echo var_dump($a)."</br>";
 echo var_dump($b)."</br>";
  echo var_dump($c)."</br>";
?>
```

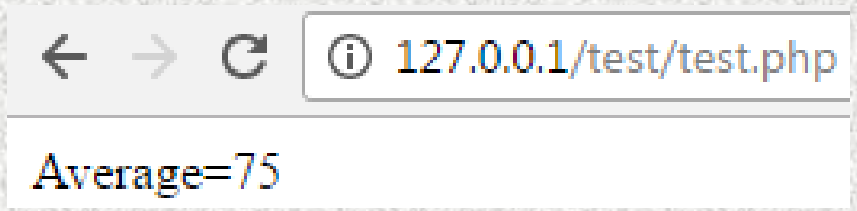


Exercise

1. Using Variable scope methedology calculate the average
   marks if marks are : maths=75, english=80, hindi=70.
2. Write a PHP script to find out the percentage of upper given marks if total marks is 300. and also find out the data type of final result.
3. Write a PHP script to calculate the 73% of 145777.
4. Write a PHP script to define a string data type and also shows
   the length of string.

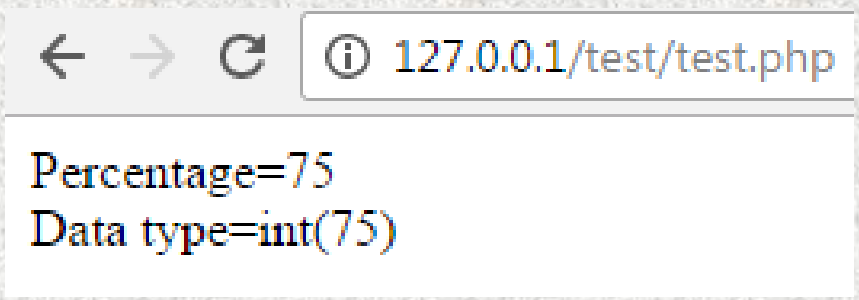
1. Using Variable scope methedology calculate the average marks if marks are : maths=75, english=80, hindi=70.

```
<?php
$maths=75;
function average()
{
    $english=80;
    static $hindi=70;
    echo "Average=".($GLOBALS['maths']+$english+$hindi)/3;
}
average();
?>
```



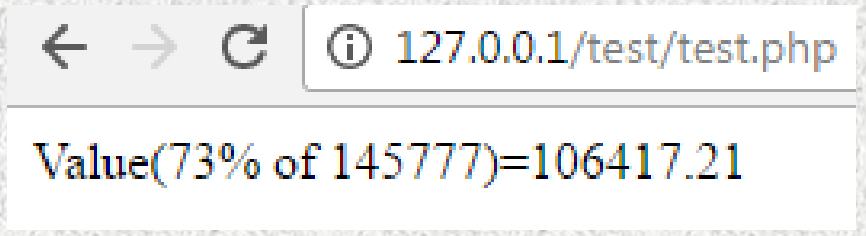
2. Write a PHP script to find out the percentage of upper given marks if total marks is 300 and also find out the data type of final result.

```
<?php
$maths=75;
function calculate()
{
        $english=80;
        static $hindi=70;
        $percentage=($GLOBALS['maths']+$english+$hindi)*100/300;
        echo "Percentage=".$percentage."<br>";
        echo "Data type=";
        echo var_dump($percentage);
}
calculate();
?>
```



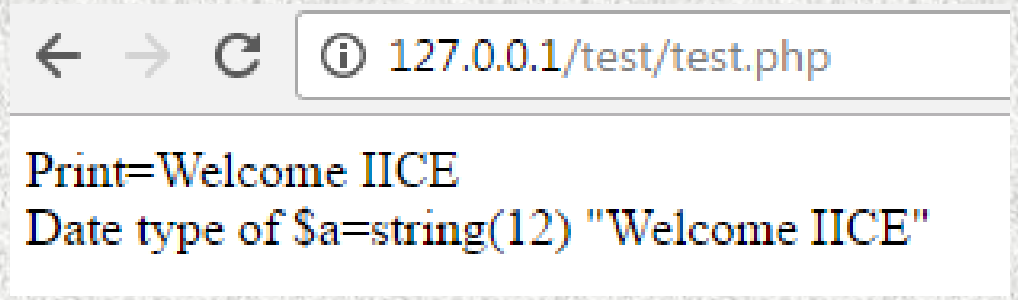
3. Write a PHP script to calculate the 73% of 145777.

```
<?php
$a=73;
$b=145777;
echo "Value(73% of 145777)=".$a*$b/100;
?>
```



4. Write a PHP script to define a string data type and also shows the length of string.

```
<?php
$a="Welcome IICE";
echo "Print=".$a;
echo "<br>Date type of \$a=";
echo var_dump($a);
?>
```



Lesson 3

String Functions

- strlen(): function returns the length of a string.
- str_word_count(): function counts the number of words in a string.
- strrev(): function reverses a string.
- strpos():function searches for a specific text within a string.
- str_replace(): function replaces some characters with some other characters in a string.

Example

```
<?php
echo strlen("Hello world!")."</br>";
echo str_word_count("Hello world!")."</br>";
echo strrev("Hello world!")."</br>";
echo strpos("Hello world!", "world!")."</br>";
echo str_replace("world", "IICE", "Hello
 world!")."</br>";
?>
```

```
12
2
!dlrow olleH
6
Hello IICE!
```

Constants

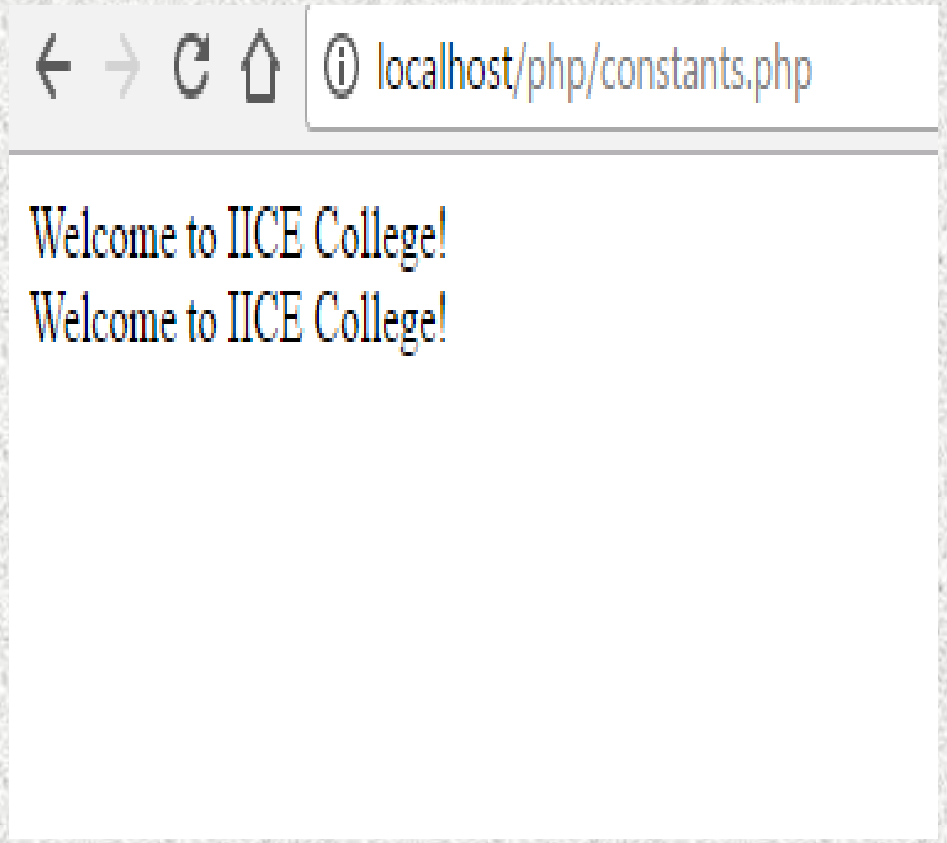
- A constant is an identifier (name) for a simple value. The value cannot be changed during the script.
- A valid constant name starts with a letter or underscore (no $ sign before the constant name).

Note: Unlike variables, constants are automatically global across the entire script.

- To create a constant, use the define() function.
- Syntax
- define(*name*, *value*, *case-insensitive*)

Example

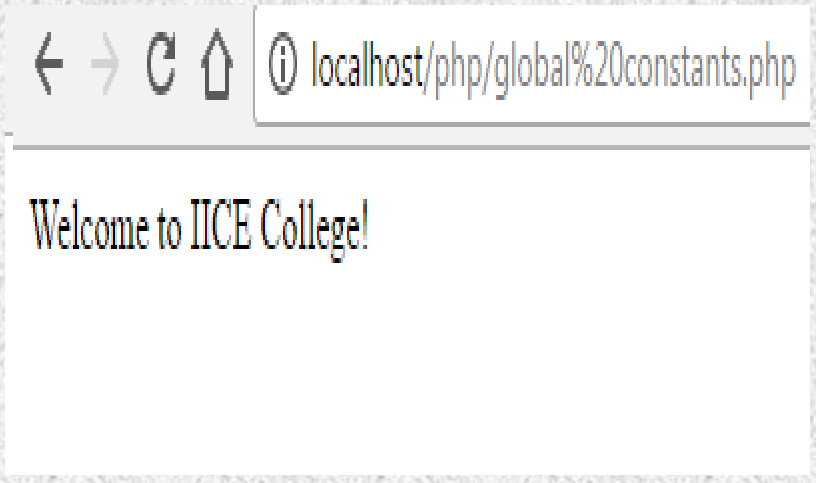
```
<?php
// case-sensitive constant name
define("a","Welcome to IICE College!");
echo a."<br/>";
//constant with a case-insensitive name
define("B", "Welcome to IICE College!", true);
echo b;
?>
```



Constants

- Constants are automatically global.

```
<?php
define("GREETING", "Welcome to IICE College!");
function myTest() {
    echo GREETING;
}
 myTest();
?>
```



Exercise

1. Write a PHP script to to apply the following function on given string-

"My First PHP script"

   i. Find out the length of string.
   ii. Find out the no. of words in string.
   iii. Reverse given string.
   iv. Find out the position of character 'i'.
   v. Replace "script" into "program".

2. Write a PHP script to define a constant
   i. a=45 in case sensitive mode.
   ii. b=34 in case insensitive mode.

1. Write a PHP script to to apply the following function on given string-

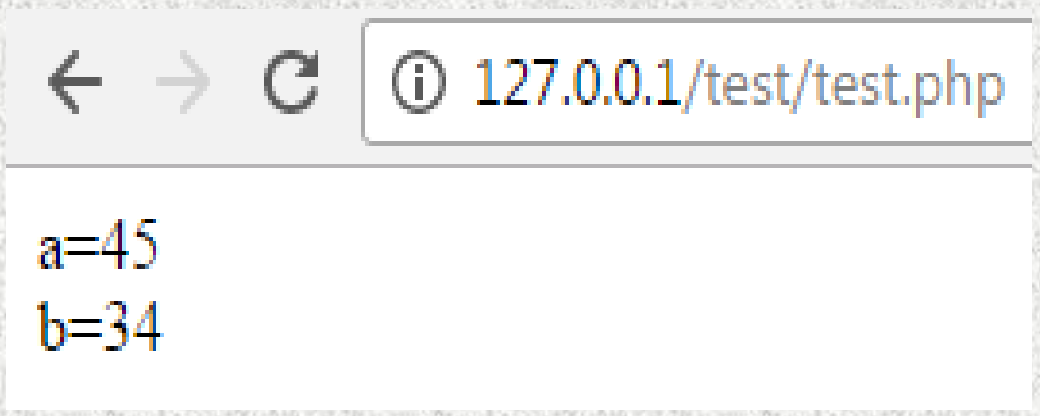
"My First PHP script"

```
19
4
tpircs PHP tsriF yM
4
My First PHP program
```

```
<?php
$a="My First PHP script";
echo strlen($a)."</br>";
echo str_word_count($a)."</br>";
echo strrev($a)."</br>";
echo strpos($a, "i")."</br>";
echo str_replace("script", "program",$a)."</br>";
?>
```

2. Write a PHP script to define a constant
i. a=45 in case sensitive mode.
ii. b=34 in case insensitive mode.

```
<?php
// case-sensitive constant name
define("a",45);
echo "a=".a."<br/>";
//constant with a case-insensitive name
define("B",34, true);
echo "b=".b;
?>
```



Lesson 4

Operators

- Arithmetic operators
- Assignment operators
- Comparison operators
- Increment/Decrement operators
- Logical operators
- String operators
- Array operators

Arithmetic Operators

Operator	Name	Example	Result
+	Addition	$x + $y	Sum of $x and $y
-	Subtraction	$x - $y	Difference of $x and $y
*	Multiplication	$x * $y	Product of $x and $y
/	Division	$x / $y	Quotient of $x and $y
%	Modulus	$x % $y	Remainder of $x divided by $y

Assignment Operators

Assignment	Same as...	Description
x = y	x = y	The left operand gets set to the value of the expression on the right
x += y	x = x + y	Addition
x -= y	x = x - y	Subtraction
x *= y	x = x * y	Multiplication
x /= y	x = x / y	Division
x %= y	x = x % y	Modulus

Comparison Operators

Operator	Name	Example	Result
==	Equal	$x == $y	Returns true if $x is equal to $y
===	Identical	$x === $y	Returns true if $x is equal to $y, and they are of the same type
!=	Not equal	$x != $y	Returns true if $x is not equal to $y
<>	Not equal	$x <> $y	Returns true if $x is not equal to $y
!==	Not identical	$x !== $y	Returns true if $x is not equal to $y, or they are not of the same type
>	Greater than	$x > $y	Returns true if $x is greater than $y
<	Less than	$x < $y	Returns true if $x is less than $y
>=	Greater than or equal to	$x >= $y	Returns true if $x is greater than or equal to $y
<=	Less than or equal to	$x <= $y	Returns true if $x is less than or equal to $y

Increment / Decrement Operators

Operator	Name	Description
++$x	Pre-increment	Increments $x by one, then returns $x
$x++	Post-increment	Returns $x, then increments $x by one
--$x	Pre-decrement	Decrements $x by one, then returns $x
$x--	Post-decrement	Returns $x, then decrements $x by one

Logical Operators

Operator	Name	Example	Result
and	And	$x and $y	True if both $x and $y are true
or	Or	$x or $y	True if either $x or $y is true
xor	Xor	$x xor $y	True if either $x or $y is true, but not both
&&	And	$x && $y	True if both $x and $y are true
||	Or	$x || $y	True if either $x or $y is true
!	Not	!$x	True if $x is not true

String Operators

Operator	Name	Example	Result
.	Concatenation	$txt1 . $txt2	Concatenation of $txt1 and $txt2
.=	Concatenation assignment	$txt1 .= $txt2	Appends $txt2 to $txt1

Array Operators

Operator	Name	Example	Result
+	Union	$x + $y	Union of $x and $y
= =	Equality	$x == $y	Returns true if $x and $y have the same key/value pairs
= = =	Identity	$x === $y	Returns true if $x and $y have the same key/value pairs in the same order and of the same types
!=	Inequality	$x != $y	Returns true if $x is not equal to $y
<>	Inequality	$x <> $y	Returns true if $x is not equal to $y
!= =	Non-identity	$x !== $y	Returns true if $x is not identical to $y

Lesson 5

if...else...elseif Statements

- **if statement** - executes some code if one condition is true

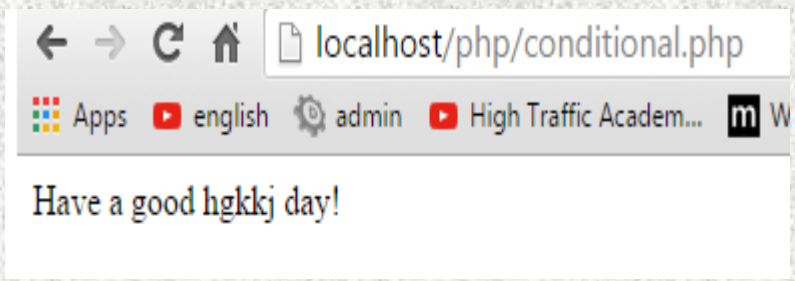
- **if...else statement** - executes some code if a condition is true and another code if that condition is false

- **if...elseif....else statement** - executes different codes for more than two conditions

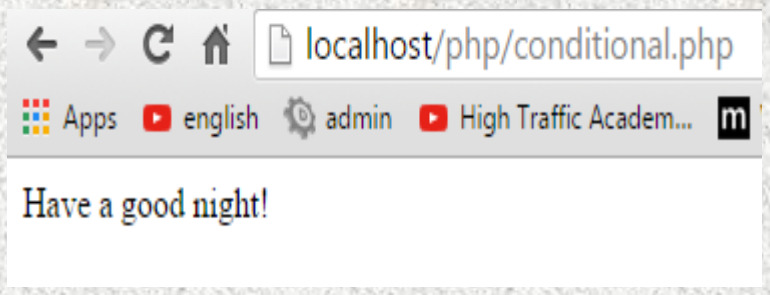
- **switch statement** - selects one of many blocks of code to be executed

Example

```
<?php
$t = 10;
if ($t < 20)
{    echo "Have a good day!";  }
?>
```

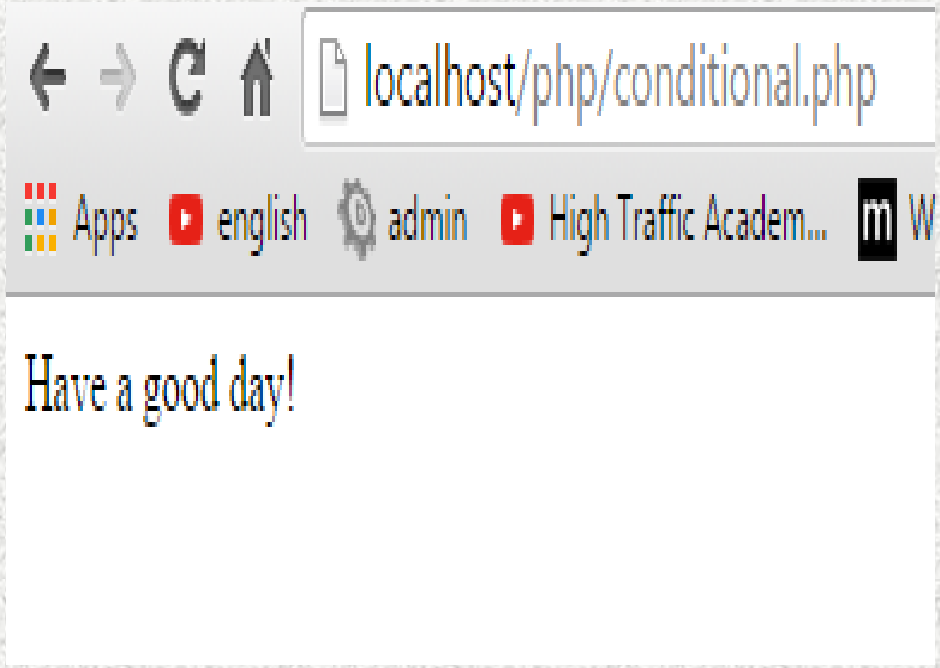



```
<?php
$t = 30;
if ($t < 20) {  echo "Have a good day!"; }
else {    echo "Have a good night!"; }
?>
```



Example

```
<?php
$t = 20;
if ($t < 10) {
    echo "Have a good morning!";
} elseif ($t < 30) {
    echo "Have a good day!";
} else {
    echo "Have a good night!";
}
?>
```

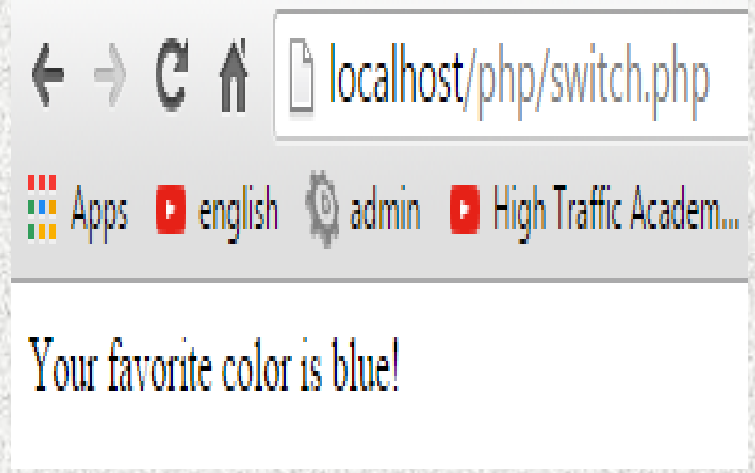


switch Statement

- First we have a single expression *n* (most often a variable), that is evaluated once.
- The value of the expression is then compared with the values for each case in the structure.
- If there is a match, the block of code associated with that case is executed.
- Use **break** to prevent the code from running into the next case automatically.
- The **default** statement is used if no match is found.

Example

```
<?php
$favcolor = "blue";
switch ($favcolor) {
    case "red":
        echo "Your favorite color is red!";       break;
    case "blue":
        echo "Your favorite color is blue!";      break;
    case "green":
        echo "Your favorite color is green!";     break;
    default:
    echo "Your favorite color is not red, blue,  green!";
}
?>
```



Exercise

1. Write a PHP script to find out the year is leap or not.
2. Write a PHP script to find out whether the a character is alphabet or not.
3. Write a PHP script to find out the larger number between three numbers.
4. Write a PHP script to create Simple Calculator using switch case.

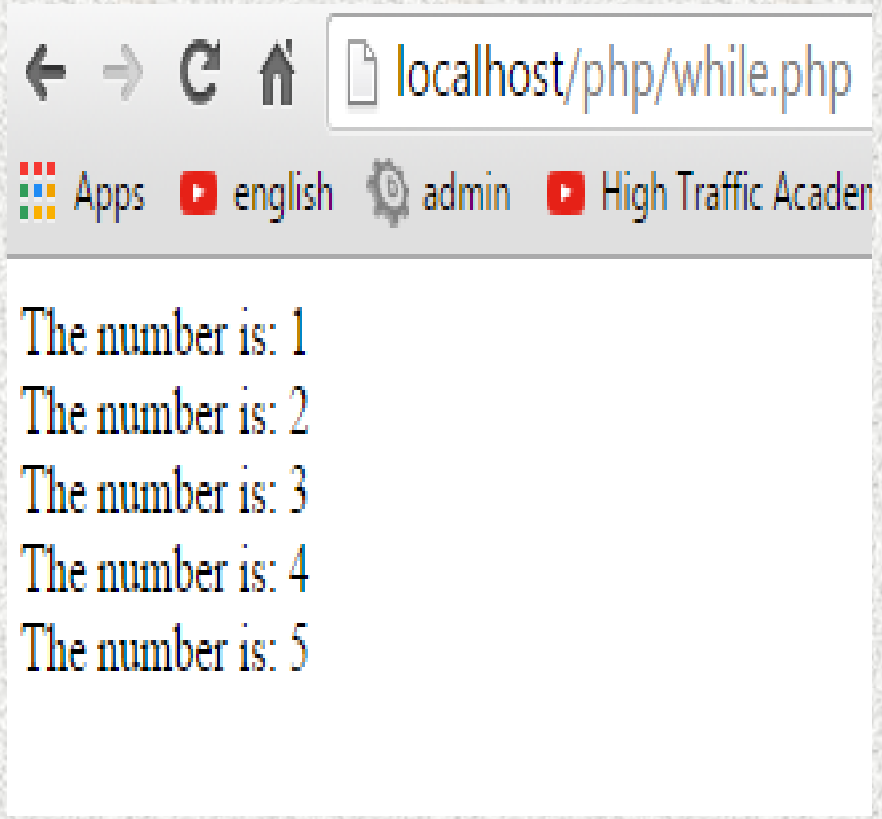
Lesson 6

Loops

- **while** - loops through a block of code as long as the specified condition is true
- **do...while** - loops through a block of code once, and then repeats the loop as long as the specified condition is true
- **for** - loops through a block of code a specified number of times
- **foreach** - loops through a block of code for each element in an array

while Loop

```
<?php
$x = 1;
while($x <= 5)
 {
    echo "The number is: $x <br>";
    $x++;
}
?>
```



do...while Loop

```
<?php
$x = 1;
do {     echo "The number is: $x <br>";
         $x++;
    } while ($x <= 5);
?>
```

The number is: 1
The number is: 2
The number is: 3
The number is: 4
The number is: 5

```
<?php
$x = 6;
do {     echo "The number is: $x <br>";
         $x++;
    } while ($x <= 5);
?>
```

The number is: 6

for Loop

Syntax

- for (*init counter; test counter; increment counter*) {
  code to be executed;
  }

- *init counter* : Initialize the loop counter value

- *test counter* : Evaluated for each loop iteration. If it evaluates to TRUE, the loop continues. If it evaluates to FALSE, the loop ends.

- *increment counter* : Increases the loop counter value

Example

```
<?php
for ($x = 0; $x <= 10; $x++) {
    echo "The number is: $x <br>";
}
?>
```

The number is: 0
The number is: 1
The number is: 2
The number is: 3
The number is: 4
The number is: 5
The number is: 6
The number is: 7
The number is: 8
The number is: 9
The number is: 10

foreach Loop

The foreach loop works only on arrays, and is used to loop through each key / value pair in an array.

Syntax

```
foreach ($array as $value)
 {
    code to be executed;
  }
```

Example

```
<?php
$colors = array("red", "green", "blue ", " yellow");
foreach ($colors as $value)
{
    echo "$value <br>";
}
?>
```

red

green

blue

yellow

Exercise

1. Write a PHP script to print alphabet from a to z.
2. Write a PHP script to print all odd no. between 0 to 100.
3. Write a PHP script to calculate factorial of given number.
4. Write a PHP script to find out Prime number.

Lesson 7

Function

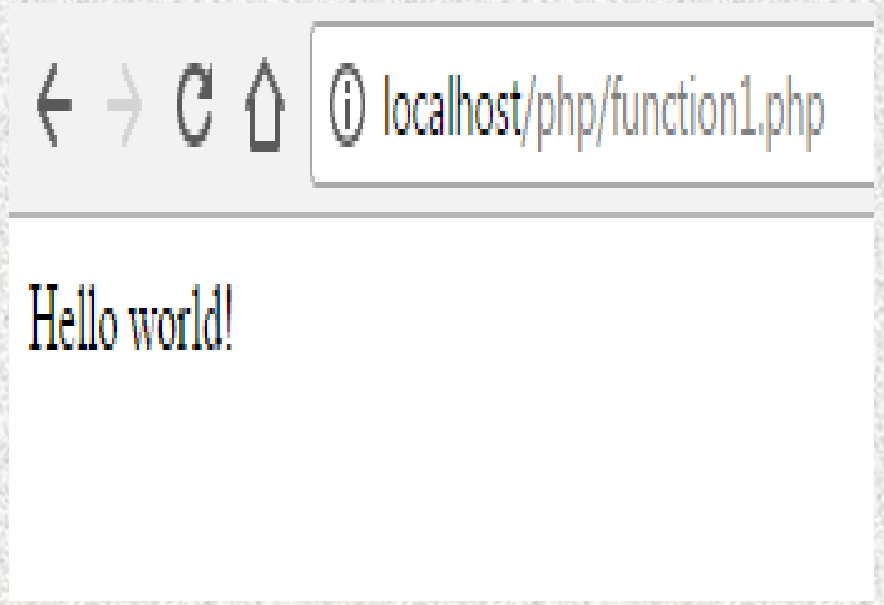
- A function is a block of statements that can be used repeatedly in a program.
- A function will not execute immediately when a page loads.
- A function will be executed by a call to the function.

Syntax:

```
function functionName () {
    code to be executed;
}
```

Example

```
<?php
function writeMsg()
 {
    echo "Hello world!";
}
writeMsg(); // call the function
?>
```



Function Arguments

Information can be passed to functions through arguments. An argument is just like a variable.

```
<?php
function familyName($fname)
 {
    echo "$fname Refsnes.<br>";
 }
familyName("Jani");
familyName("Hege");
familyName("Stale");
familyName("Kai Jim");
familyName("Borge");
?>
```

Jani Refsnes.
Hege Refsnes.
Stale Refsnes.
Kai Jim Refsnes.
Borge Refsnes.

Default Argument

If we call the function without arguments it takes the default value as argument.

```
<?php
function setHeight($minheight = 50) {
    echo "The height is : $minheight <br>";
}
setHeight(350);
setHeight(); //the default value of 50
setHeight(135);
setHeight(80);
?>
```

```
The height is : 350
The height is : 50
The height is : 135
The height is : 80
```

Returning values

```
<?php
function sum($x, $y)
{
    $z = $x + $y;
    return $z;
}
echo "5 + 10 = " . sum(5, 10) . "<br>";
echo "7 + 13 = " . sum(7, 13) . "<br>";
echo "2 + 4 = " . sum(2, 4);
?>
```

$5 + 10 = 15$
$7 + 13 = 20$
$2 + 4 = 6$

Exercise

1. Write a PHP script to find out the cube of any number using function.
2. Write a PHP script to find out the no. is even or odd using function.
3. Write a PHP script to display array element using function.
4. Write a PHP script to sum numbers passing arguments.

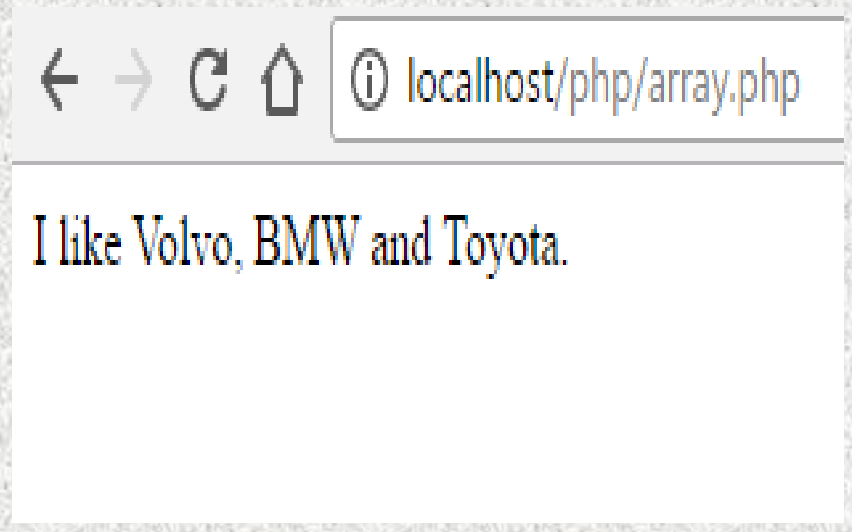
Lesson 8

Arrays

- An array stores multiple values in one single variable.
- An array is a special variable, which can hold more than one value at a time.
- array() function is used to create an array.
- There are three types of arrays:
  - **Indexed arrays** - Arrays with a numeric index
  - **Associative arrays** - Arrays with named keys
  - **Multidimensional arrays** - Arrays containing one or more arrays

Indexed Array

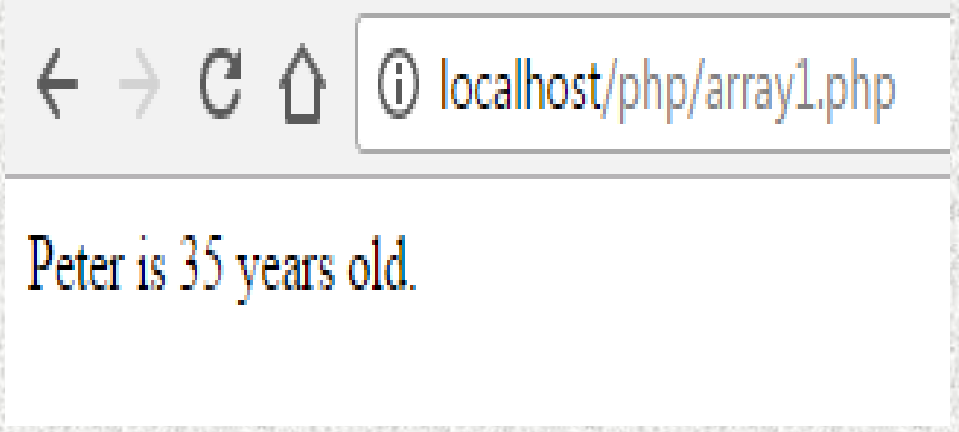
```
<?php
$cars = array("Volvo", "BMW",
"Toyota");
echo "I like " . $cars[0] . ", " . $cars[1] .
" and " . $cars[2] . ".";
?>
```



Associative Arrays

Associative arrays are arrays that use named keys that you assign to them.

```
<?php
$age = array("Peter"=>"35", "Ben"=>"37");
echo "Peter is " . $age['Peter'] . " years
old.";
?>
```



Sort Functions For Arrays

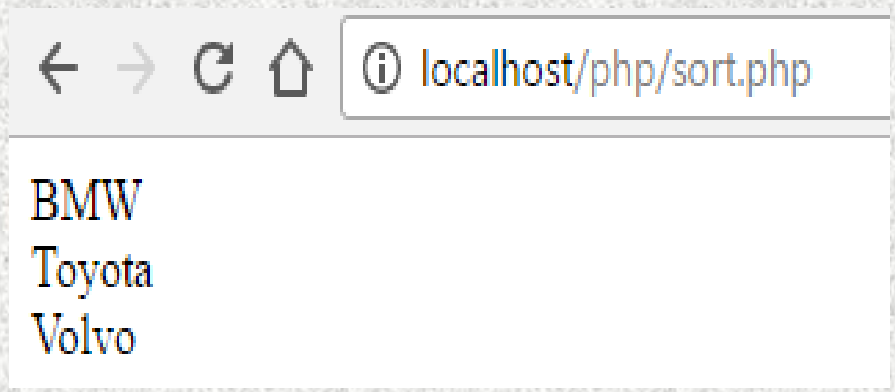
PHP array sort functions:

- sort() - sort arrays in ascending order
- rsort() - sort arrays in descending order
- asort() - sort associative arrays in ascending order, according to the value
- ksort() - sort associative arrays in ascending order, according to the key
- arsort() - sort associative arrays in descending order, according to the value
- krsort() - sort associative arrays in descending order, according to the key

Example

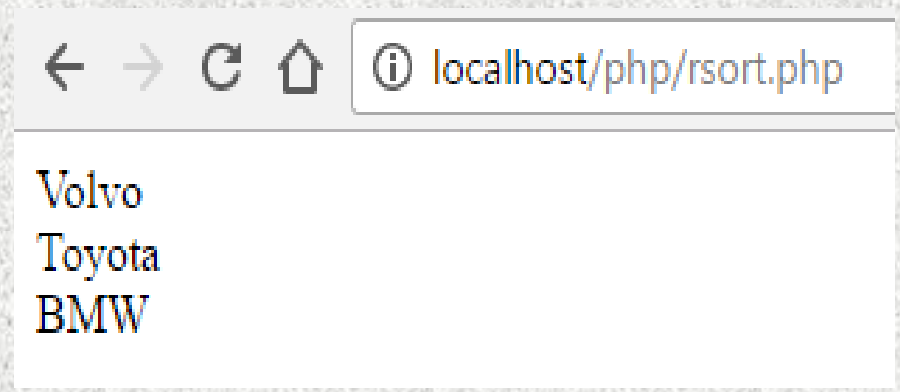
sort()

```
<?php
$cars = array("Volvo", "BMW", "Toyota");
sort($cars);
$clength = count($cars);
for($x = 0; $x < $clength; $x++) {
    echo $cars[$x];
    echo "<br>";
}
?>
```



rsort()

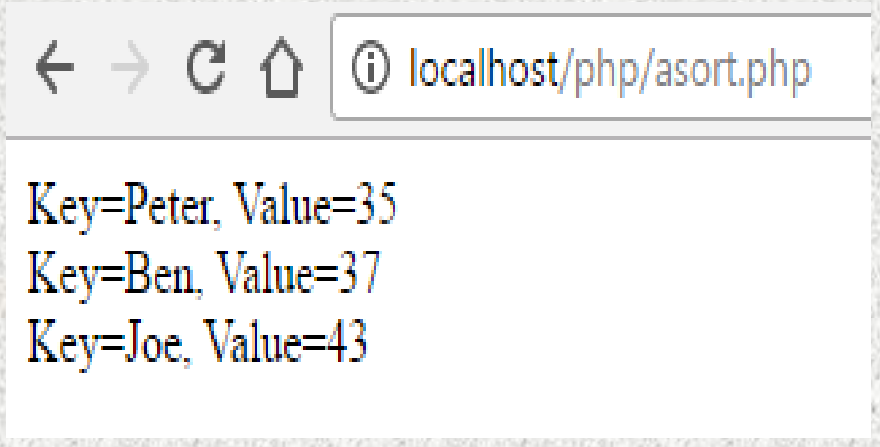
```
<?php
$cars = array("Volvo", "BMW", "Toyota");
rsort($cars);

$clength = count($cars);
for($x = 0; $x < $clength; $x++) {
    echo $cars[$x];
    echo "<br>";
}
?>
```



Example

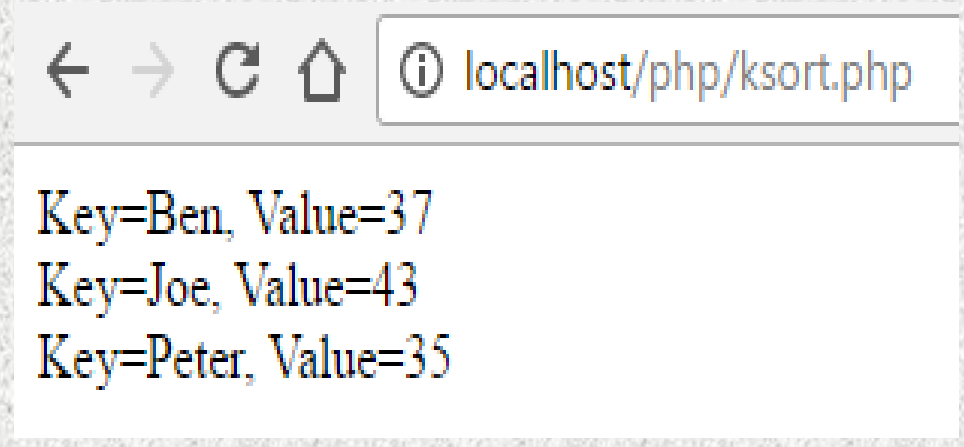
asort()

```
<?php
$age = array("Peter"=>"35", "Ben"=>"37",
        "Joe"=>"43");
asort($age);
foreach($age as $x => $x_value) {
 echo "Key=" . $x . ", Value=" . $x_value;
 echo "<br>";
}
?>
```



ksort()

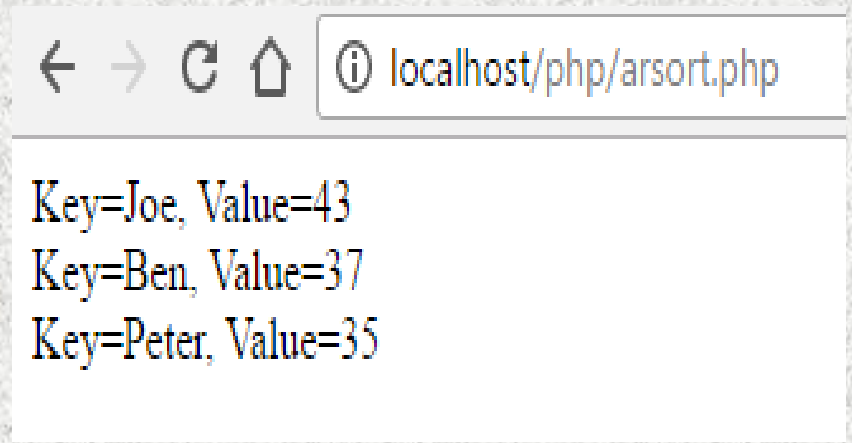
```
<?php
$age = array("Peter"=>"35", "Ben"=>"37",
        "Joe"=>"43");
ksort($age);
foreach($age as $x => $x_value) {
 echo "Key=" . $x . ", Value=" . $x_value;
 echo "<br>";
}
?>
```



Example

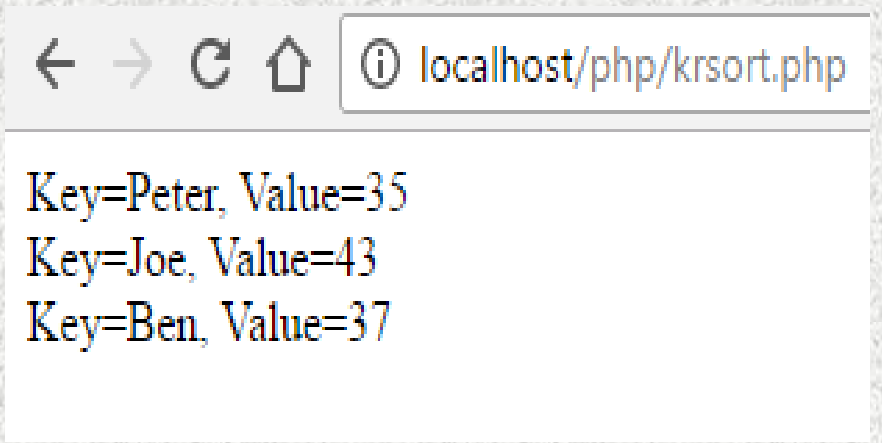
arsort()

```
<?php
$age = array("Peter"=>"35", "Ben"=>"37",
        "Joe"=>"43");
arsort($age);
foreach($age as $x => $x_value) {
    echo "Key=" . $x . ", Value=" . $x_value;
    echo "<br>";
}
?>
```



krsort()

```
<?php
$age = array("Peter"=>"35", "Ben"=>"37",
"Joe"=>"43");
krsort($age);
foreach($age as $x => $x_value) {
    echo "Key=" . $x . ", Value=" . $x_value;
    echo "<br>";
}
?>
```



Exercise

1. Write a PHP script to apply the following function on an array.

   sort(),rsort(),asort(),ksort(),arsort(),krsort()

2. Write a PHP script to defined the indexed and associative arrays.

Lesson 9

Server Installation

- Install server to execute PHP code and to create database and run sql code.
- Servers like :- xampp, wamp etc.
  - **32 bit :-** https://www.apachefriends.org/xampp-files/5.5.38/xampp-win32-5.5.38-3-VC11-installer.exe
  - **64 bit :-** https://www.apachefriends.org/xampp-files/5.5.38/xampp-linux-x64-5.5.38-3-installer.run

MySQL Database

- MySQL is a database system used on the web.
- MySQL is a database system that runs on a server.
- MySQL uses standard SQL.

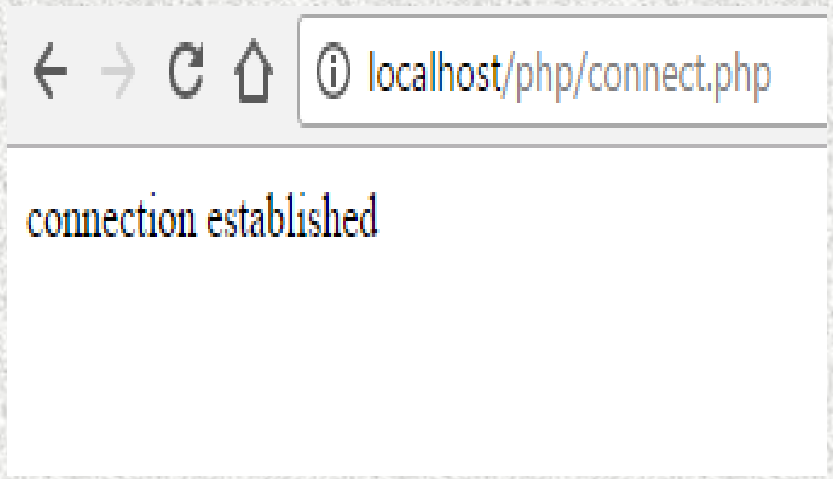
Connect to MySQL

Syntax:

```
<?php
        $servername = "localhost";
        $username = "username";
        $password = "password";
        mysql_connect($servername,$username,$password);
?>
```

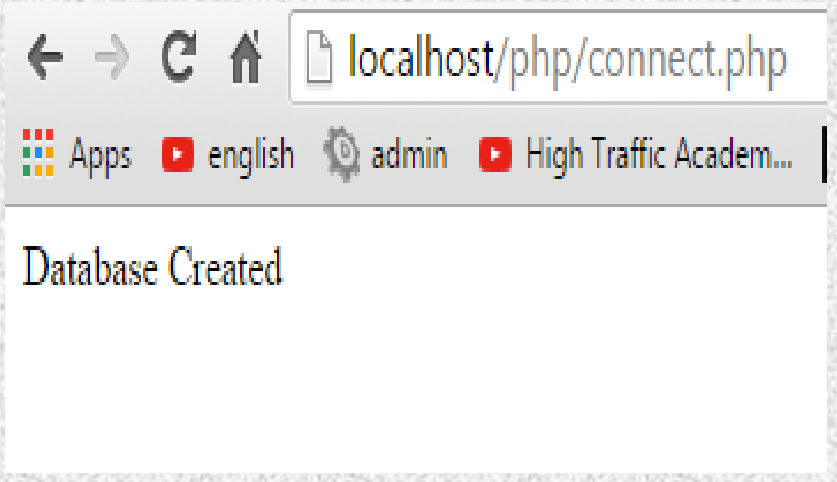
Example

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
mysql_connect($servername,$username,$password);
echo 'connection established';
?>
```



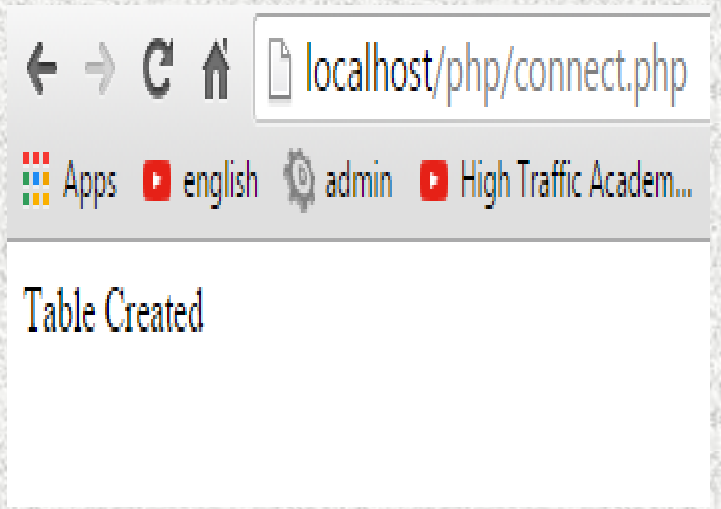
Create MySQL Database

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$con=mysql_connect($servername,$username,$password);
$a=$con.mysql_query("CREATE DATABASE database");
if($a)
echo 'Database Created';
?>
```



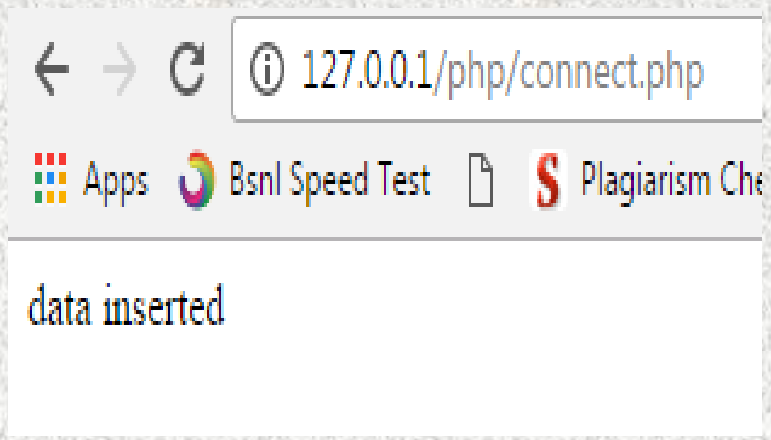
Create MySQL Table

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$con=mysql_connect($servername,$username,$password);
$con.mysql_query("CREATE DATABASE database");
mysql_select_db("database");
$sql = "CREATE TABLE MyGuests (
id int(6),firstname varchar(30))";
$b=mysql_query($sql);
if($b)
echo 'Table Created';
?>
```



Insert Operation

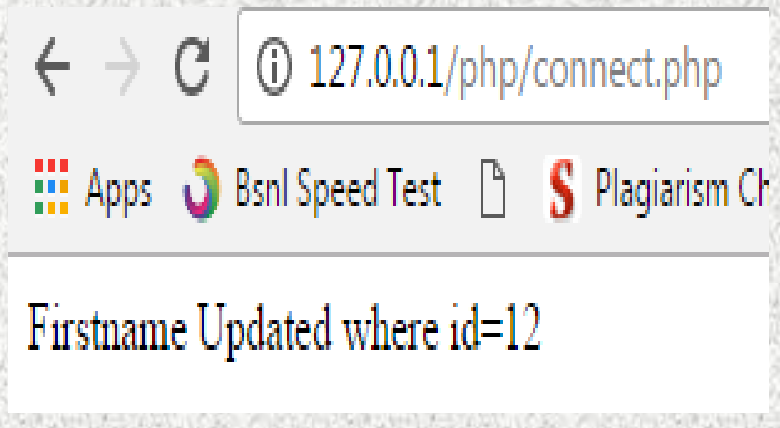
```
<?php
$con=mysql_connect("localhost","root","");
mysql_select_db("database1");
$id=12;
$firstname="John";
$c=mysql_query("insert into MyGuests values('$id','$firstname')");
if($c)
        echo 'data inserted';
?>
```



			id	firstname
☐	✎	✕	12	John

Update Operation

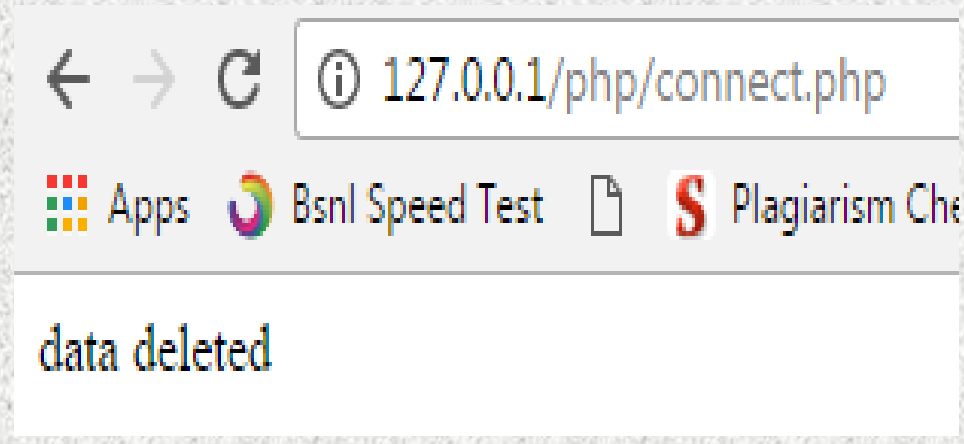
```
<?php
$con=mysql_connect("localhost","root","");
mysql_select_db("database1");
$id=12;
$firstname="Harry";
$c=mysql_query("update MyGuests set firstname='$firstname' where
    id='$id'");
if($c)
    echo 'Firstname Updated where id='.$id;
?>
```



			id	firstname
☐	✎	✕	12	Harry

Delete Data

```
<?php
$con=mysql_connect("localhost","root","");
mysql_select_db("database1");
$id=12;
$c=mysql_query("delete from MyGuests where id='$id'");
if($c)
    echo 'data deleted';
?>
```



Exercise

1. Write a PHP script to create a database and table in Mysql server.
2. Write a PHP script to insert following values in table.

Name	Father_Name	Age	City
Ram Kumar	Mr. Vikas	18	Bikaner
Kishan	Mr. Abhay	17	Jaisalmer
Manohar	Mr. Ramesh	19	Jodhpur
Vijay	Mr. Kailash	20	Alwar

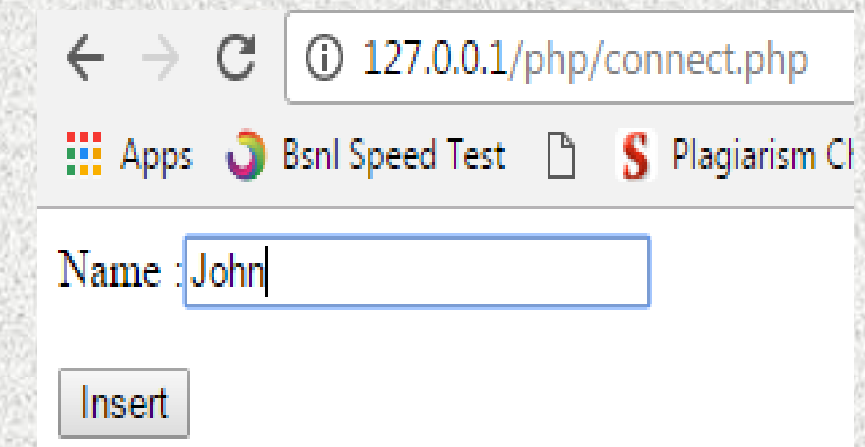
3. Write a PHP script to update the Kishan age=19 and Vijay age=18.
4. Write a PHP script to delete the record of Vijay from table.

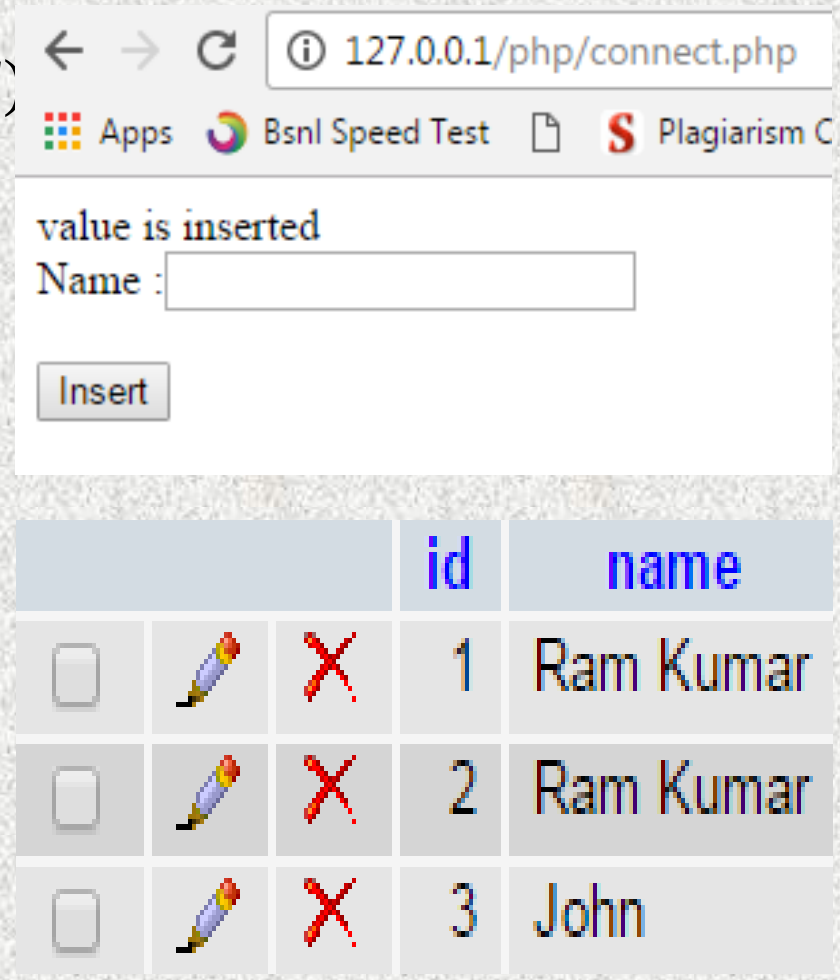
Lesson 10

Form Handling with MySQL

Insert Operation

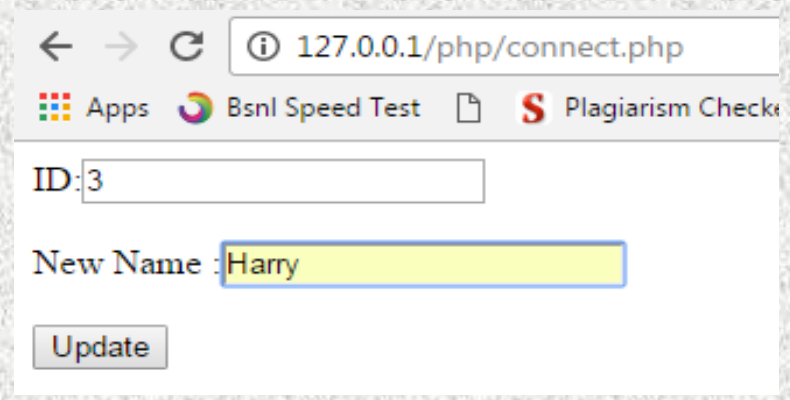
```
<?php
$name="";
$con=mysql_connect("localhost","root","");
mysql_select_db("database1");
if ($_SERVER["REQUEST_METHOD"] == "POST")
{
$name=$_POST['name'];
$c=mysql_query("insert into record(name)values('$name')
if($c)
    echo 'value is inserted';
}
echo '<form method="post" action="connect.php">
Name :<input type="text" name="name"/><br><br>
<input type="submit" value="Insert"/></form>';
?>
```

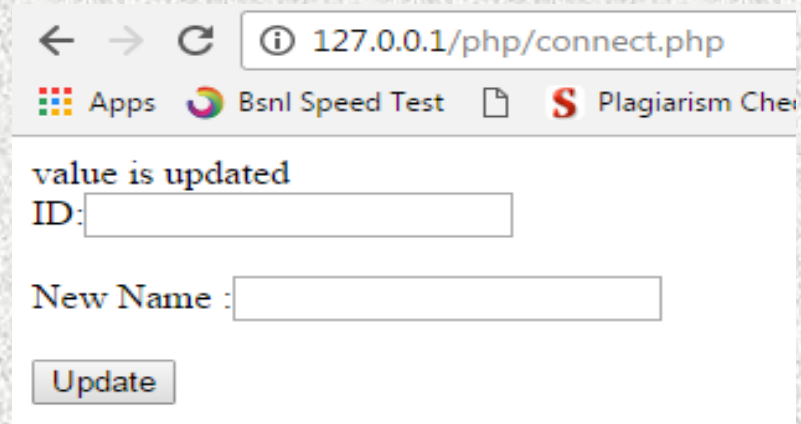




Update Operation

```
<?php
$name = $id = "";
$con=mysql_connect("localhost","root","");
mysql_select_db("database1");
if ($_SERVER["REQUEST_METHOD"] == "POST")
{
    $id=$_POST['id'];
    $name=$_POST['name'];
    $c=mysql_query("update record set name='$name' where
    id='$id'");
if($c)
    echo 'value is updated';
}
echo '<form method="post" action="connect.php">
    ID:<input type="number" name="id"/><br><br>
    New Name :<input type="text" name="name"/><br><br>
    <input type="submit" value="Update"/></form>';
?>
```

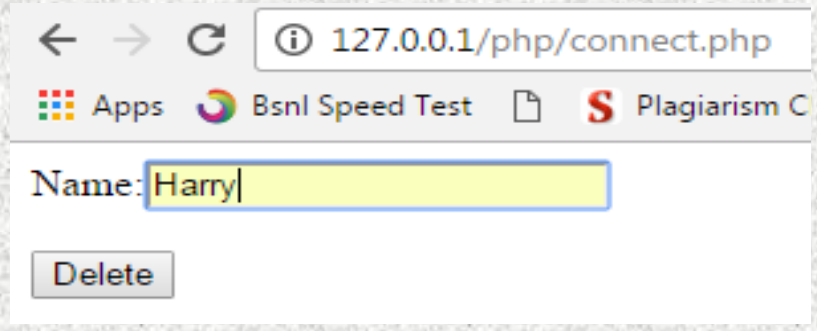


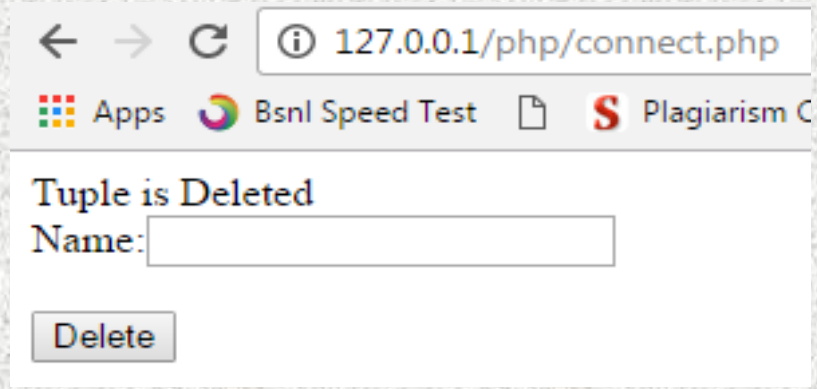


			id	name
☐	✎	✗	1	Ram Kumar
☐	✎	✗	2	Ram Kumar
☐	✎	✗	3	Harry

Delete Operation

```
<?php
$name = "";
$con=mysql_connect("localhost","root","");
mysql_select_db("database1");
if ($_SERVER["REQUEST_METHOD"] == "POST")
{
    $name=$_POST['name'];
    $c=mysql_query("delete from record where
    name='$name'");
if($c)
    echo 'Tuple is Deleted';
}
echo '<form method="post" action="connect.php">
Name:<input type="name" name="name"/><br><br>
<input type="submit" value="Delete"/></form>';
?>
```

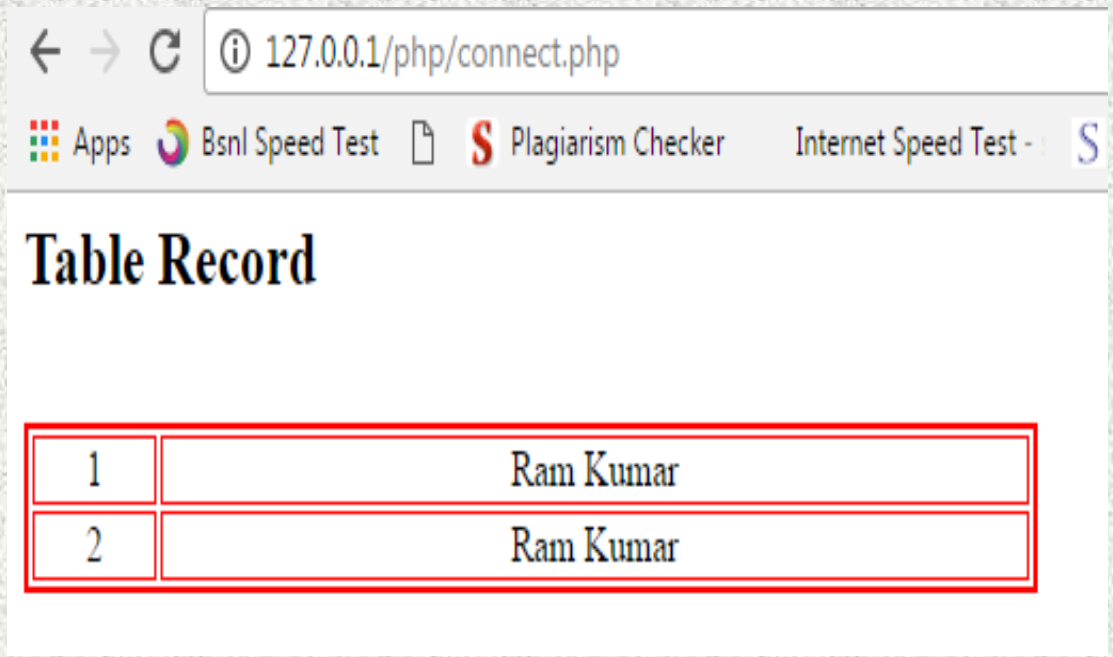




			id	name
☐	✎	✕	1	Ram Kumar
☐	✎	✕	2	Ram Kumar

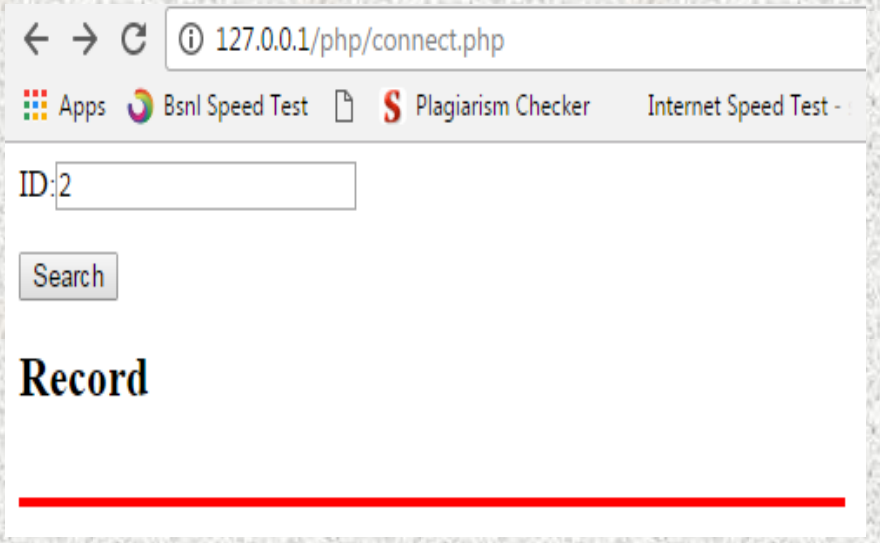
Display Table Data

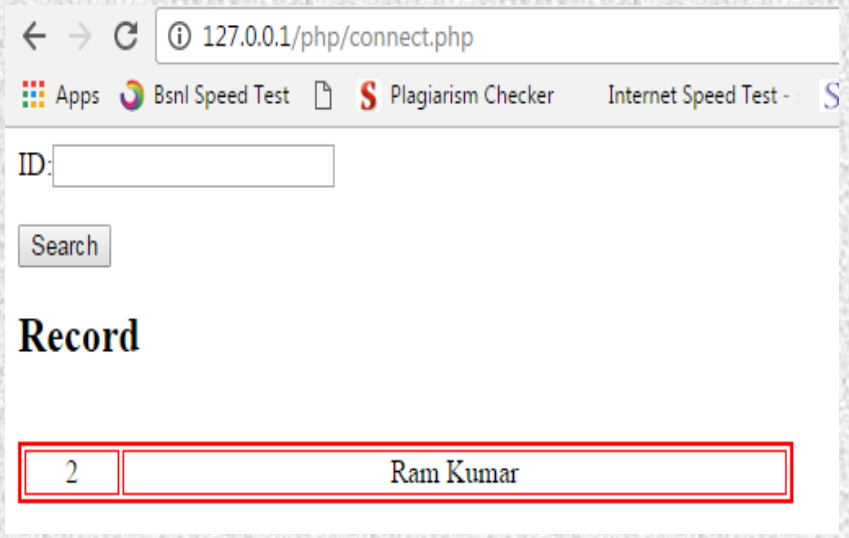
```
<?php
$name = "";
$con=mysql_connect("localhost","root","");
mysql_select_db("database1");
$c=mysql_query("select * from record");
echo '<h2>Table Record</h2><br><table
    height="auto" width="30%" border="2"
    bordercolor="red">';
while($row=mysql_fetch_array($c))
{
    echo '<tr>
    <td align="center">'.$row['id'].'</td>
    <td align="center">'.$row['name'].'</td>
    </tr>';
}
echo '</table>';
?>
```



Selection Operation

```
<?php
echo '<form method="post" action="connect.php">
ID:<input type="number" name="id"/><br><br>
<input type="submit" value="Search"/></form>';
$con=mysql_connect("localhost","root","");
mysql_select_db("database1");
$id=$_POST['id'];
$c=mysql_query("select * from record");
echo '<h2>Record</h2><br><table height="auto" width="30%" border="2"
    bordercolor="red">';
while($row=mysql_fetch_array($c))
{   if($row['id']==$id)
    echo '<tr><td align="center">'.$row['id'].'</td>
    <td align="center">'.$row['name'].'</td></tr>';
}
echo '</table>';
?>
```

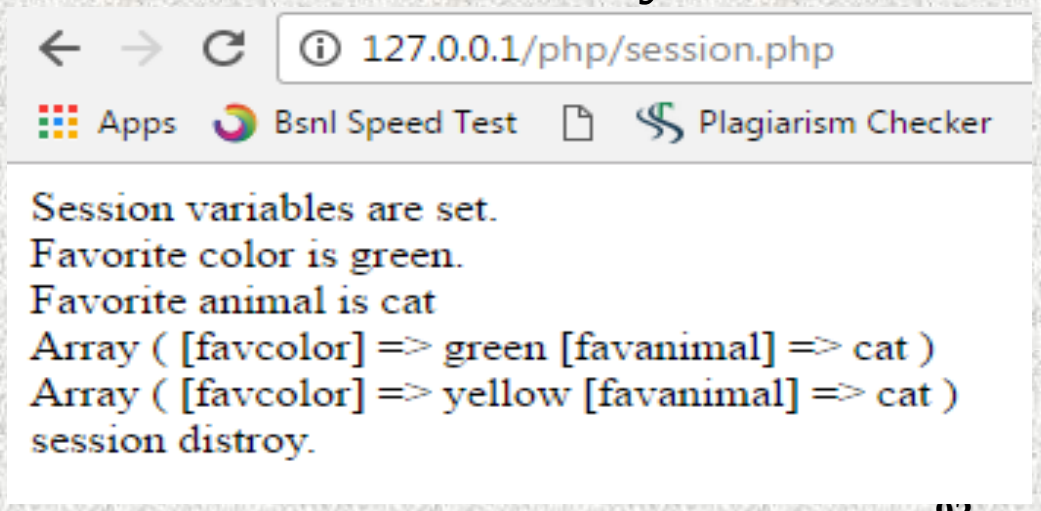




Sessions

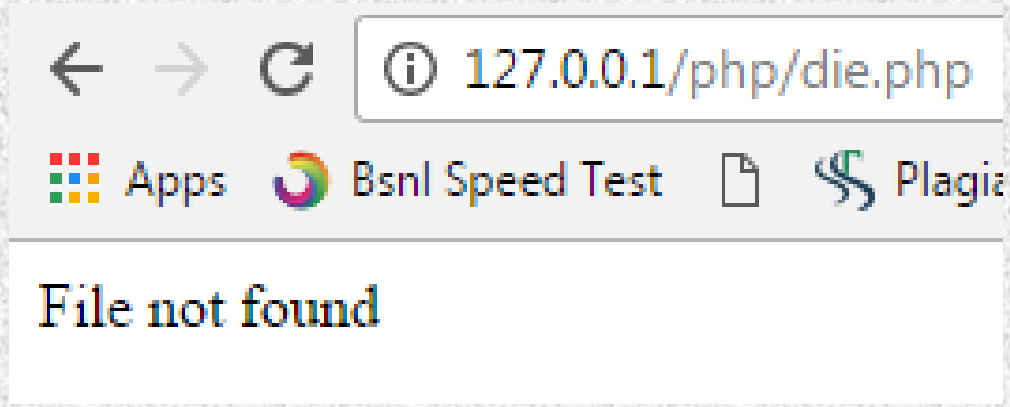
A session is a way to store information (in variables) to be used across multiple pages.

```
<?php
// Start the session
session_start();
// Set session variables
$_SESSION["favcolor"] = "green";
$_SESSION["favanimal"] = "cat";
echo "Session variables are set.";
/* Echo session variables that were set
 on previous page*/
echo "<br>Favorite color is " .
$_SESSION["favcolor"] . ".<br>";
echo "Favorite animal is " .
$_SESSION["favanimal"] . "<br>";
print_r($_SESSION);
//Modify Session
$_SESSION["favcolor"] = "yellow";
echo "<br>";
print_r($_SESSION);
// remove all session variables
session_unset();
// destroy the session
session_destroy();
echo "<br>session distroy."
?>
```



die() function

```
<?php
if(!file_exists("welcome.txt")) {
die("File not found");
} else {
$file=fopen("welcome.txt","r");
}
?>
```



Exception Handling

Exception Handling done by 3 functions :

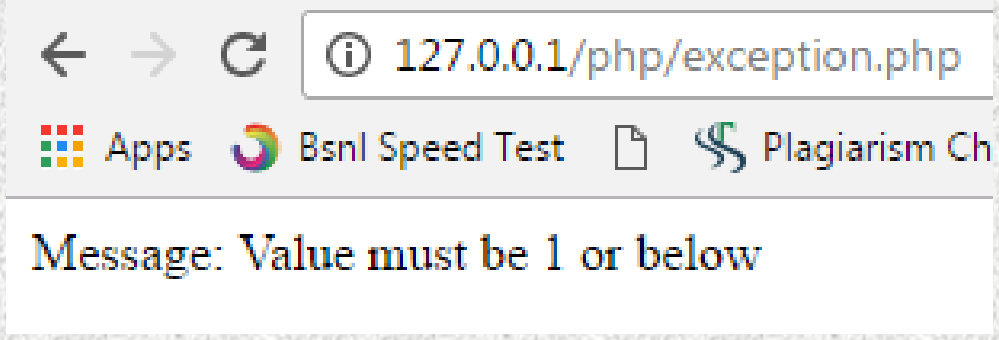
try - A function using an exception should be in a "try" block. If the exception does not trigger, the code will continue as normal. However if the exception triggers, an exception is "thrown"

throw - This is how you trigger an exception. Each "throw" must have at least one "catch"

catch - A "catch" block retrieves an exception and creates an object containing the exception information

Example

```
<?php
//create function with an exception
function checkNum($number) {
  if($number>1) {
    throw new Exception("Value must be 1 or below");
  }
  return true;    }
//trigger exception in a "try" block
try {
  checkNum(2);
  //If the exception is thrown, this text will not be shown
  echo 'If you see this, the number is 1 or below';      }
//catch exception
catch(Exception $e) {
  echo 'Message: ' .$e->getMessage();
}
?>
```



Exercise

1. Write a PHP script to create a form for any account creation and save details in database.

2. Write a PHP script to create a login page for upear created account with password verification from database.
3. Write a PHP scripted for logout after login into account.

Note:- Use session for Q.2 and Q.3 implementation.

Lesson 11

Form Validation Functions

- htmlspecialchars() This prevents attackers from exploiting the code by injecting HTML or Javascript code in forms.
- trim() Strip unnecessary characters (extra space, tab, newline) from the user input data.
- stripslashes() Remove backslashes (\) from the user input data.
- empty() check whether input field value is null or not.
- preg_match() Searches a string for pattern, returning true if the pattern exists, and false otherwise.
- filter_var() use to validate email address.

Form Validation

- Validation Rules are used to apply the restriction on the user input.

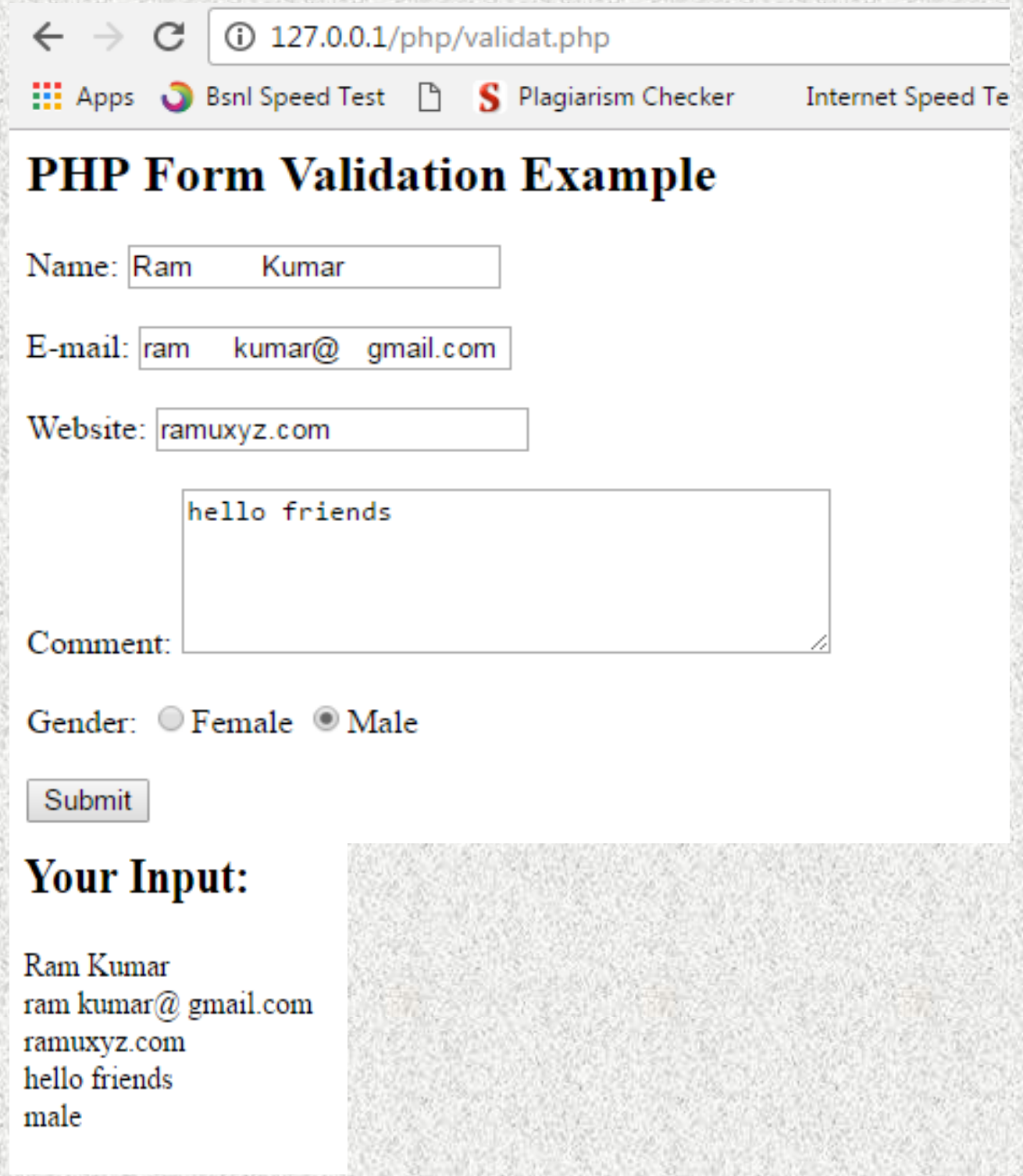
- General Rules of Form Validation:
  - pass all variables through PHP's htmlspecialchars() function.
  - Apply trim() function.
  - Apply stripslashes() function.

Example

```
<?php
// define variables and set to empty values
$name = $email = $gender = $comment = $website = "";
if ($_SERVER["REQUEST_METHOD"] == "POST")
{
  $name = test_input($_POST["name"]);
  $email = test_input($_POST["email"]);
  $website = test_input($_POST["website"]);
  $comment = test_input($_POST["comment"]);
  $gender = test_input($_POST["gender"]);
}
function test_input($data) {
  $data = trim($data);
  $data = stripslashes($data);
  $data = htmlspecialchars($data);
  return $data; }
```

```
echo '<h2>PHP Form Validation Example</h2>
<form method="post"
    action='.htmlspecialchars($_SERVER["PHP_SELF"]).'>
  Name: <input type="text" name="name"><br><br>
  E-mail: <input type="text" name="email"><br><br>
  Website: <input type="text" name="website"> <br><br>
  Comment: <textarea name="comment" rows="5"
    cols="40"></textarea><br><br>
  Gender:
  <input type="radio" name="gender" value="female">Female
  <input type="radio" name="gender" value="male">Male
  <br><br>
  <br><br>
  <input type="submit" name="submit" value="Submit">
</form>';
echo "<h2>Your Input:</h2>";
echo $name."<br>“. $email. "<br>”.$website."<br>”.$comment;
echo "<br>“. $gender;          ?>
```

Output


127.0.0.1/php/validat.php
Apps
Bsnl Speed Test
Plagiarism Checker
Internet Speed Te
PHP Form Validation Example
Name: Ram Kumar
E-mail: ram kumar@ gmail.com
Website: ramuxyz.com
Comment: hello friends
Gender: Female Male
Submit
Your Input:
Ram Kumar
ram kumar@ gmail.com
ramuxyz.com
hello friends
male

Display Error Messages

- The user tries to submit the form without filling out the required fields display error message.

```
<?php
$nameErr=$emailErr=$genderErr="";
$name=$email=$gender="";
if
($_SERVER["REQUEST_METHOD"]=="POST"
) {
  if (empty($_POST["name"])) {
    $nameErr="Name is required";
  } else {
    $name=test_input($_POST["name"]);
  } if (empty($_POST["email"])) {
    $emailErr="Email is required";
  } else {
    $email=test_input($_POST["email"]);
  }if (empty($_POST["gender"])) {
    $genderErr="Gender is required";
  } else {
    $gender=test_input($_POST["gender"]);
  }          }
```

```
function test_input($data) {
  $data=trim($data);
  $data=stripslashes($data);
  $data=htmlspecialchars($data);          return $data;
   } echo '<h2>PHP Form Validation Example</h2>
<p><span class="error">* required field.</span></p>
<form method="post"
    action='.htmlspecialchars($_SERVER["PHP_SELF"]).'>
  Name: <input type="text" name="name">
  <span class="error">* '.$nameErr.'</span><br><br>
  E-mail: <input type="text" name="email">
  <span class="error">* '.$emailErr.'</span><br><br>
  Gender:
  <input type="radio" name="gender" value="female">Female
  <input type="radio" name="gender" value="male">Male
  <span class="error">*'.$genderErr.'</span><br><br>
  <input type="submit" name="submit" value="Submit">
</form>';              echo "<h2>Your Input:</h2>";
echo $name. "<br>". $email."<br>". $gender;      ?>
```

Output

Screen 1

127.0.0.1/php/validat1.php

Apps Bsnl Speed Test Plagiarism Checker

PHP Form Validation Example

* required field.

Name: Ram Kumar *

E-mail: *

Gender: Female Male *

Submit

Your Input:

Screen 2

127.0.0.1/php/validat1.php

Apps Bsnl Speed Test Plagiarism Checker Internet S

PHP Form Validation Example

* required field.

Name: *

E-mail: * Email is required

Gender: Female Male *

Submit

Your Input:

Ram Kumar

male

Lesson 12

Validate E-mail and URL

- To check an email address is well-formed is to use filter_var() function.
- To check a URL address is well-formed is to use preg_match() function.

Email validation syntax

```
$email = test_input($_POST["email"]);
    if (!filter_var($email,
    FILTER_VALIDATE_EMAIL))
{
     $emailErr = "Invalid email format";
}
```

URL Validation syntax

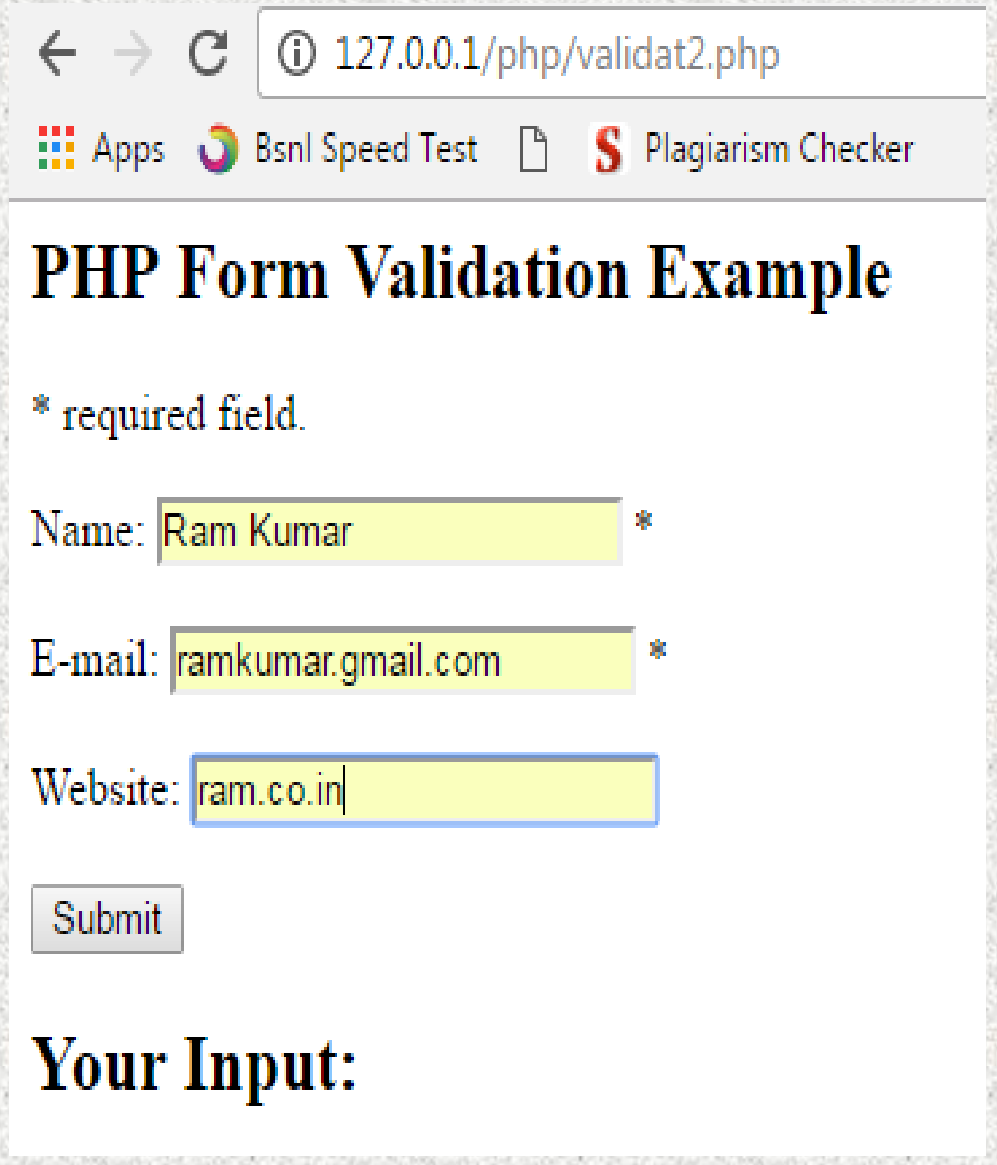
```
$website = test_input($_POST["website"]);
if(!preg_match("/\b(?:(?:https?|ftp):\/\/|ww
    w\.)[-a-z0-9+&@#\/%?=~_|!:,.;]*[-a-z0-
    9+&@#\/%=~_|]/i",$website))
{
     $websiteErr = "Invalid URL";
}
```

Example

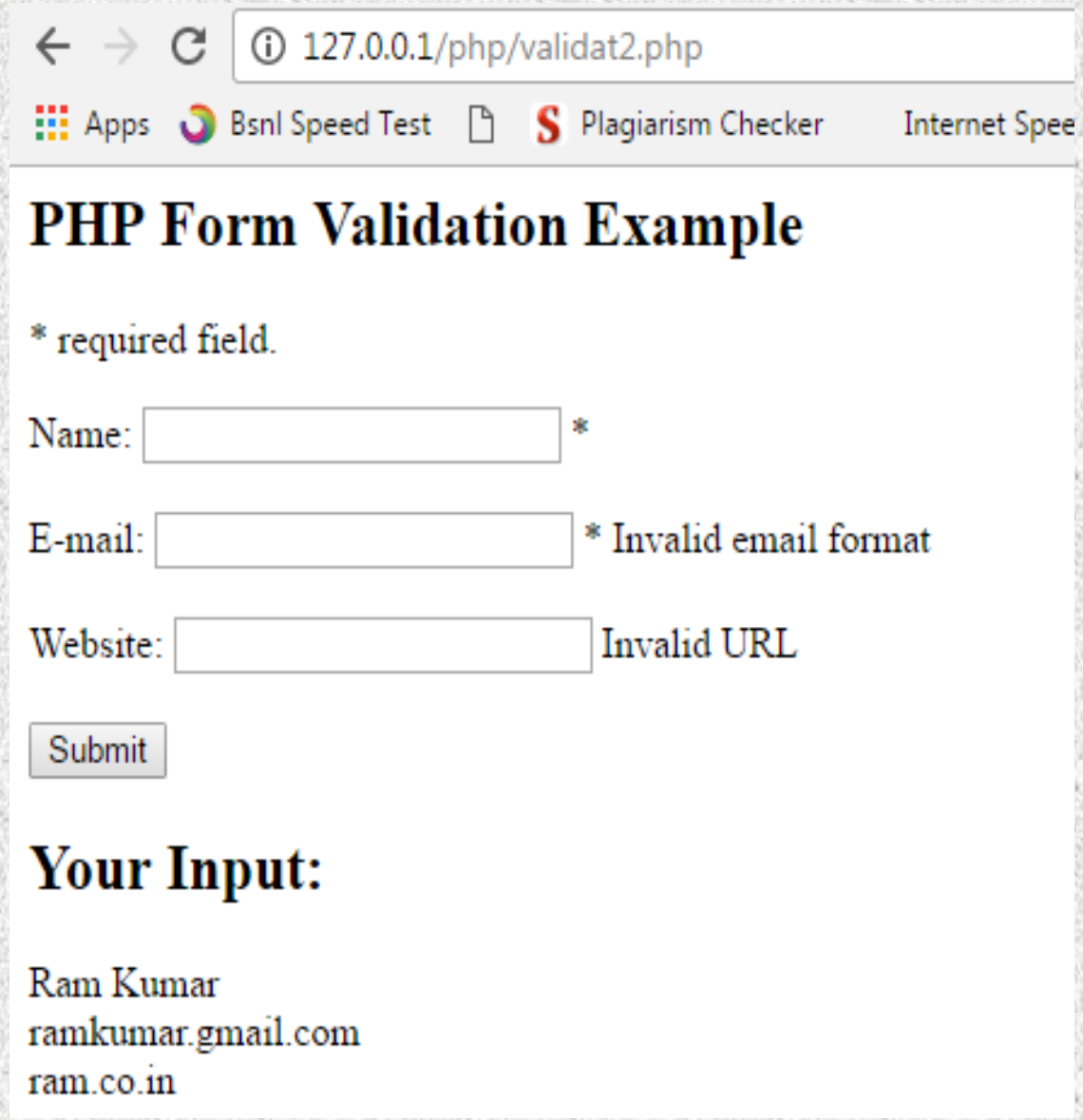
```
<?php
$nameErr = $emailErr = $genderErr = $websiteErr = "";
$name = $email = $gender = "";
if ($_SERVER["REQUEST_METHOD"] == "POST") {
  if (empty($_POST["name"])) {
    $nameErr = "Name is required";
  } else {  $name = test_input($_POST["name"]);
if (!preg_match("/^[a-zA-Z ]*$/",$name)) {
      $nameErr = "Only letters and white space allowed";
    } } if (empty($_POST["email"])) {
    $emailErr = "Email is required";
  } else { $email = test_input($_POST["email"]);
if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
      $emailErr = "Invalid email format";
    } }  if (empty($_POST["website"])) {
    $website = "";
} else {  $website = test_input($_POST["website"]);
if (!preg_match("/\b(?:(?:https?|ftp):\/\/|www\.)[-a-z0-
      9+&@#\/%?=~_|!:,.;]*[-a-z0-
      9+&@#\/%=~_|]/i",$website)) {
      $websiteErr = "Invalid URL“;    }     }     }
function test_input($data) {
  $data = trim($data);
  $data = stripslashes($data);
  $data = htmlspecialchars($data);  return $data;
}
echo '<h2>PHP Form Validation Example</h2>
<p><span class="error">* required field.</span></p>
<form method="post"
      action='.htmlspecialchars($_SERVER["PHP_SELF"]).'
      >  Name: <input type="text" name="name">
  <span class="error">* '.$nameErr.'</span><br><br>
  E-mail: <input type="text" name="email">
  <span class="error">* '.$emailErr.'</span><br><br>
  Website: <input type="text" name="website">
  <span class="error">'.$websiteErr.'</span><br><br>
  <input type="submit" name="submit" value="Submit">
     </form>';
echo "<h2>Your Input:</h2>";
echo $name."<br>".$email."<br>".$website;   ?>
```

Output

Screen 1

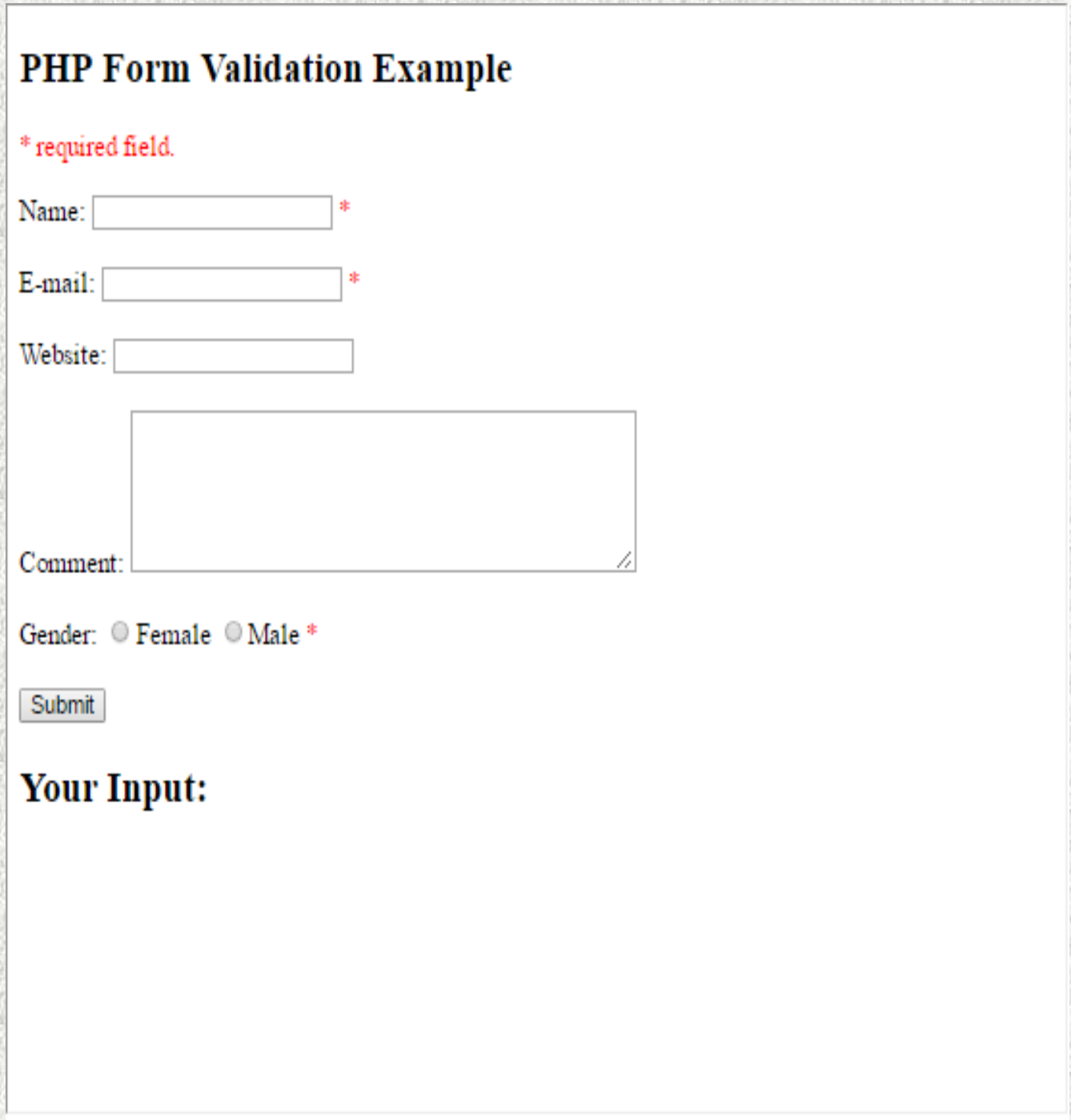


Screen 2



Exercise

1. Design a complete form as shown below with applying all validation rules.



```
<html><head><style>
.error {color: #FF0000;}</style>
</head><body>  <?php
// define variables and set to empty values
$nameErr = $emailErr = $genderErr = $websiteErr = "";
$name = $email = $gender = $comment = $website = "";
if ($_SERVER["REQUEST_METHOD"] == "POST") {
  if (empty($_POST["name"])) {
    $nameErr = "Name is required";
  } else {    $name = test_input($_POST["name"]);
    // check if name only contains letters and whitespace
    if (!preg_match("/^[a-zA-Z ]*$/",$name)) {
      $nameErr = "Only letters and white space allowed";
    }  }  if (empty($_POST["email"])) {
   $emailErr = "Email is required";
  } else {  $email = test_input($_POST["email"]);
    // check if e-mail address is well-formed
if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
     $emailErr = "Invalid email format";     }  }
if (empty($_POST["website"])) {
   $website = "";
  } else {
   $website = test_input($_POST["website"]);
   // check if URL address syntax is valid (this regular
     expression also allows dashes in the URL)
   if (!preg_match("/\b(?:(?:https?|ftp):\/\/|www\.)[-a-z0-
     9+&@#\/%?=~_|!:,.;]*[-a-z0-
     9+&@#\/%=~_|]/i",$website)) {
     $websiteErr = "Invalid URL";    }  }
 if (empty($_POST["comment"])) {
   $comment = "";
  } else {    $comment = test_input($_POST["comment"]);}
  if (empty($_POST["gender"])) {
    $genderErr = "Gender is required";
  } else {    $gender = test_input($_POST["gender"]);  }
}
function test_input($data) {  $data = trim($data);
  $data = stripslashes($data);  $data = htmlspecialchars($data);
  return $data;     }     ?>
```

Cont..

```
<h2>PHP Form Validation Example</h2>
<p><span class="error">* required
    field.</span></p>
<form method="post" action="<?php echo
    htmlspecialchars($_SERVER["PHP_SELF"]);?>
    ">
  Name: <input type="text" name="name"
    value="<?php echo $name;?>">
  <span class="error">* <?php echo
    $nameErr;?></span><br><br>
  E-mail: <input type="text" name="email"
    value="<?php echo $email;?>">
  <span class="error">* <?php echo
    $emailErr;?></span>
  <br><br>
  Website: <input type="text" name="website"
    value="<?php echo $website;?>">
  <span class="error"><?php echo
    $websiteErr;?></span>
  <br><br>
  Comment: <textarea name="comment" rows="5"
    cols="40"><?php echo $comment;?></textarea>
  <br><br>Gender:
  <input type="radio" name="gender" <?php if
    (isset($gender) && $gender=="female") echo
    "checked";?> value="female">Female
  <input type="radio" name="gender" <?php if
    (isset($gender) && $gender=="male") echo
    "checked";?> value="male">Male
  <span class="error">* <?php echo
    $genderErr;?></span><br><br>
  <input type="submit" name="submit"
    value="Submit">  </form>
<?php
echo "<h2>Your Input:</h2>";
echo $name."<br>“. $email. "<br>“. $website;
echo "<br>“. $comment. "<br>";
echo $gender;       ?>
</body></html>
```

Lesson 13

File Handling in PHP

Opening a file Reading a file Writing a file Closing a file

Mode	Purpose
r	Opens the file for reading only. Places the file pointer at the beginning of the file.
r+	Opens the file for reading and writing. Places the file pointer at the beginning of the file.
w	Opens the file for writing only. Places the file pointer at the beginning of the file. and truncates the file to zero length. If files does not exist then it attempts to create a file.
w+	Opens the file for reading and writing only. Places the file pointer at the beginning of the file. and truncates the file to zero length. If files does not exist then it attempts to create a file.
a	Opens the file for writing only. Places the file pointer at the end of the file. If files does not exist then it attempts to create a file.
a+	Opens the file for reading and writing only. Places the file pointer at the end of the file. If files does not exist then it attempts to create a file.

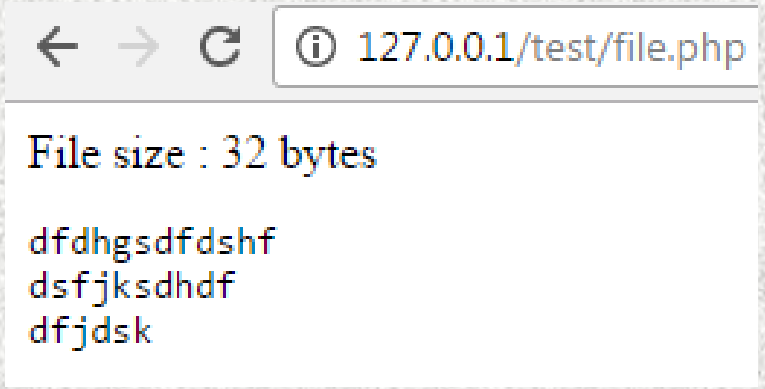
Functions

- Open a file using **fopen()** function.
- Get the file's length using **filesize()** function.
- Read the file's content using **fread()** function.
- Close the file with **fclose()** function.

Reading a file

Once a file is opened using fopen() function it can be read with a function called fread(). This function requires two arguments. These must be the file pointer and the length of the file expressed in bytes.

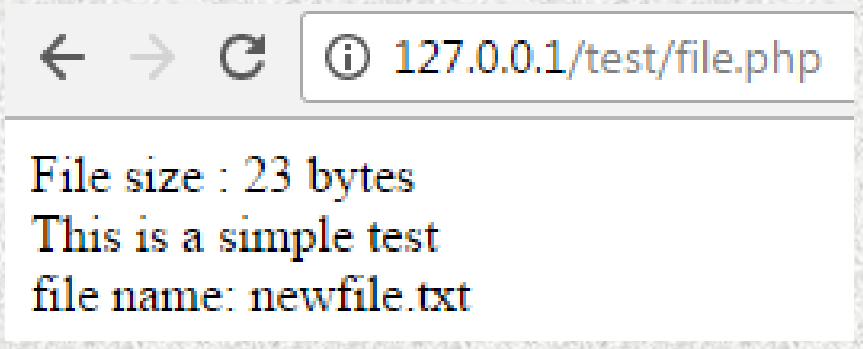
```
<?php
  $filename = "tmp.txt";
  $file = fopen( $filename, "r" );
  if( $file == false )
    {
        echo ( "Error in opening file" );
        exit();
    }
  $filesize = filesize( $filename );
  $filetext = fread( $file, $filesize );
  fclose( $file );
      echo ( "File size : $filesize bytes" );
  echo ( "<pre>$filetext</pre>" );
?>
```



Writing a file

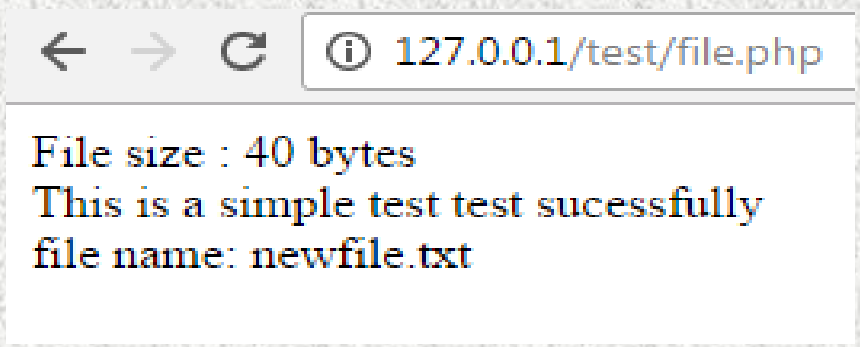
A new file can be written or text can be appended to an existing file using the PHP fwrite() function. This function requires two arguments specifying a file pointer and the string of data that is to be written.

```
<?php
  $filename = "newfile.txt";
  $file = fopen( $filename, "w" );
  if( $file == false )
{           echo ( "Error in opening new file" );
             exit();                }
fwrite( $file, "This is  a simple test\n" );
$file = fopen( $filename, "r" );
      if( $file == false ) {
        echo ( "Error in opening file" );
        exit();                    }
     $filesize = filesize( $filename );
     $filetext = fread( $file, $filesize );
     fclose( $file );
     echo ( "File size : $filesize bytes<br>" );
     echo ( "$filetext<br>" );
     echo("file name: $filename<br>");
  ?>
```



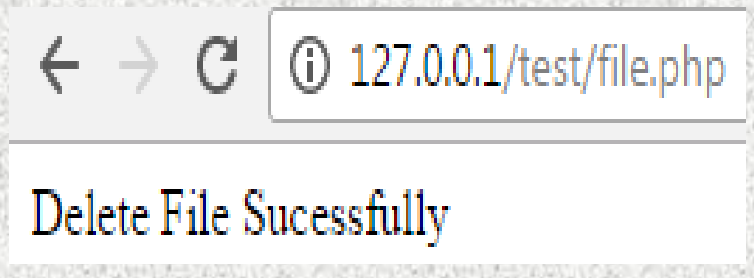
Append Operation

```
<?php
   $filename = "newfile.txt";
   $file = fopen( $filename, "a" );
      if( $file == false ) {
      echo ( "Error in opening new file" );
      exit();
   }
   fwrite( $file, "test sucessfully\n" );
    $filename = "newfile.txt";
    $file = fopen( $filename, "r" );
  if( $file == false )          {
        echo ( "Error in opening file" );
        exit();
   }
    $filesize = filesize( $filename );
    $filetext = fread( $file, $filesize );
     fclose( $file );
      echo ( "File size : $filesize bytes<br>" );
      echo ( "$filetext<br>" );
      echo("file name: $filename<br>");
  ?>
```



Delete Operation

```
<?php
$filename = "newfile.txt";
$a=unlink($filename);
if($a)
{
 echo "Delete File Sucessfully";
 }
  ?>
```



Exercise

1. Write a PHP script to create a new file "test.txt"
   content:- Hello, This is my first PHP file creation script.

2. Write a PHP script to read the above file.

3. Write a PHP script to append the following content in the above created file.
   content :- I learn File Handling in PHP

4. Write a PHP script to delete the above created file.

Lesson 14

Object Oriented Concepts

Class – This is a programmer-defined data type, which includes local functions as well as local data. You can think of a class as a template for making many instances of the same kind (or class) of object.

Object – An individual instance of the data structure defined by a class. You define a class once and then make many objects that belong to it. Objects are also known as instance.

Member Variable – These are the variables defined inside a class. This data will be invisible to the outside of the class and can be accessed via member functions. These variables are called attribute of the object once an object is created.

Member function – These are the function defined inside a class and are used to access object data.

Inheritance – When a class is defined by inheriting existing function of a parent class then it is called inheritance. Here child class will inherit all or few member functions and variables of a parent class.

Cont..

Object Oriented Concepts

Parent class – A class that is inherited from by another class. This is also called a base class or super class.

Child Class – A class that inherits from another class. This is also called a subclass or derived class.

Polymorphism – This is an object oriented concept where same function can be used for different purposes. For example function name will remain same but it make take different number of arguments and can do different task.

Overloading – a type of polymorphism in which some or all of operators have different implementations depending on the types of their arguments. Similarly functions can also be overloaded with different implementation.

Data Abstraction – Any representation of data in which the implementation details are hidden (abstracted).

Object Oriented Concepts

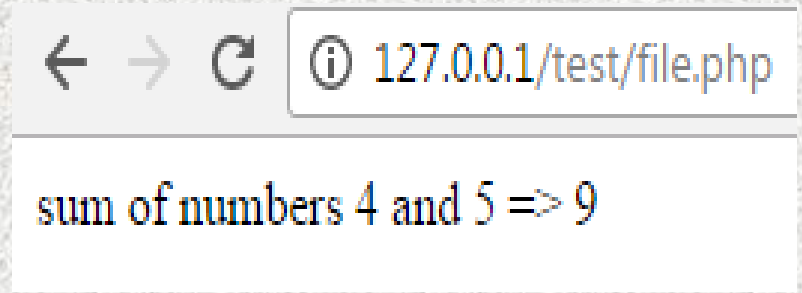
Encapsulation – refers to a concept where we encapsulate all the data and member functions together to form an object.

Constructor – refers to a special type of function which will be called automatically whenever there is an object formation from a class.

Destructor – refers to a special type of function which will be called automatically whenever an object is deleted or goes out of scope.

Class Implementation

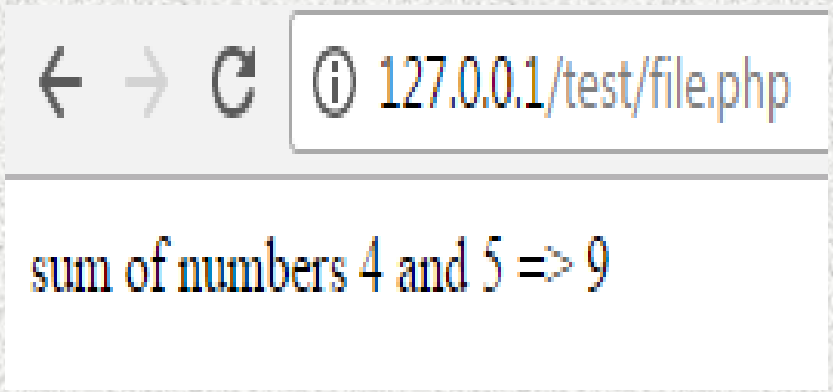
```
<?php
 class sum
 {
 var $a;
 var $b;
 var $s;
 function addition($c,$d)
 {
        $a=$c;
        $b=$d;
        $s=$a+$b;
        echo "sum of numbers ".$a." and ".$b ." => ".$s;
}
}
$s= new sum;
$s->addition(4,5);
    ?>
```



Constructor Functions

Constructor Functions are special type of functions which are called automatically whenever an object is created. So we take full advantage of this behaviour, by initializing many things through constructor functions.

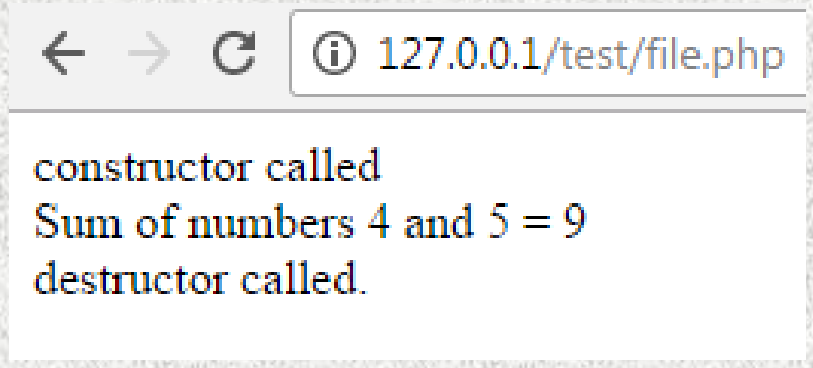
```
<?php
 class sum
 {
 var $a;
 var $b;
 var $s;
function __construct($a,$b)
 {
$this->a=$a;
$this->b=$b;
$s=$a+$b;
            echo ("sum of numbers ".$a." and ".$b ." => ".$s.PHP_EOL);
            }        }
 $s= new sum(4,5);
     ?>
```



Destructor

Like a constructor function we can define a destructor function using function **__destruct()**. We can release all the resources with-in a destructor.

```
<?php
class sum
{
var $a;
var $b;
var $s;
function __construct($a,$b)
{
        echo "constructor called<br>";
$this->a=$a;
$this->b=$b;
$s=$a+$b;
        echo ("Sum of numbers ".$a." and ".$b ." = ".$s.PHP_EOL);
        }
        function __destruct()
        {       echo "<br>destructor called.“;       }
}
$s= new sum(4,5);
    ?>
```



Inheritance

Inheritance is the concept in which when we create a new class then we can access the data members and member functions from base class to that class.

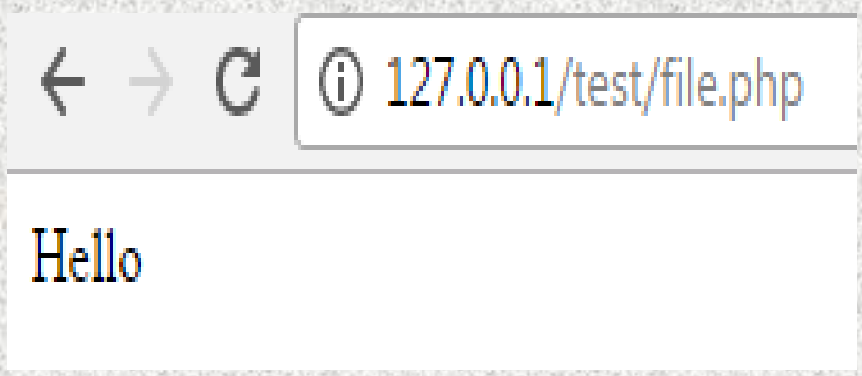
The idea of inheritance implements the **is a** relationship.

Access Control :

Access	public	protected	private
Same class	yes	yes	yes
Derived classes	yes	yes	no
Outside classes	yes	no	no

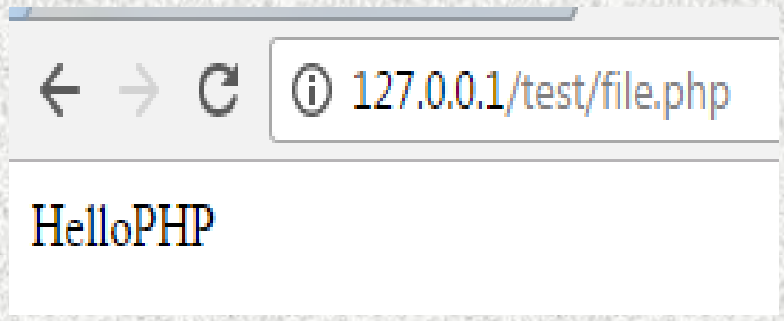
Simple Inheritance

```
<?php
class A{
        public $a='Hello';
  }
class B extends A {
   function show() {
      echo $this->a;
            }
}
$b = new B;
echo($b->show());
?>
```



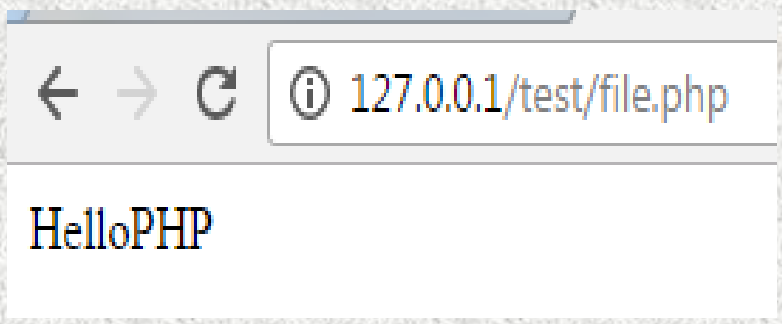
Multilevel Inheritance

```
<?php
class A{
        public $a='Hello';
  }
class B extends A {
   public $b='PHP‘;              }
class C extends B{
function show() {
      echo $this->a.$this->b;       }
}
$c = new C;
echo($c->show());
?>
```



Multiple Inheritance

```
<?php
interface A{    public function display1();     }
interface B {   public function display2();       }
class C implements A, B
{
public function display1()      {
 echo "function display1 of interface A is called";    }
public function display2(){
echo "<br>function display2 of interface B is called";         }
}
$c = new C;
echo($c->display1());
echo($c->display2());
?>
```

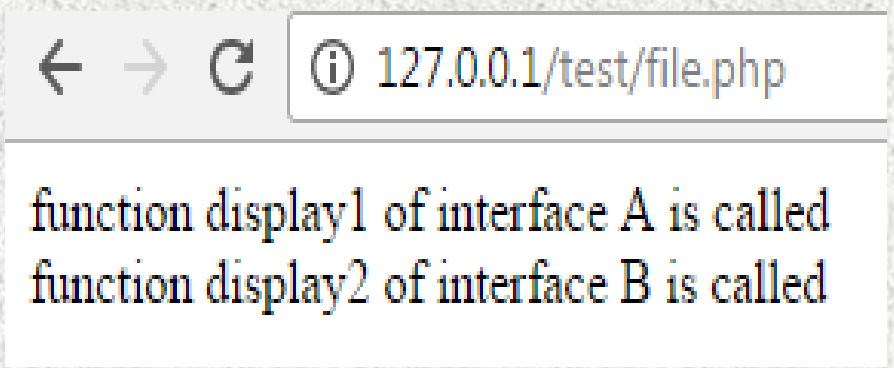


Note :- Multiple Inheritance implements using Interfaces.

Function Overriding

Function definitions in child classes override definitions with the same name in parent classes.

```
<?php
class A{
   function show() {
      return "Hello";      }
}
class B extends A {
   function show() {
      return "!! Good Morning !!";       }
}
$a = new A;
$b = new B;
echo($a->show());
echo($b->show());
?>
```



Exercise