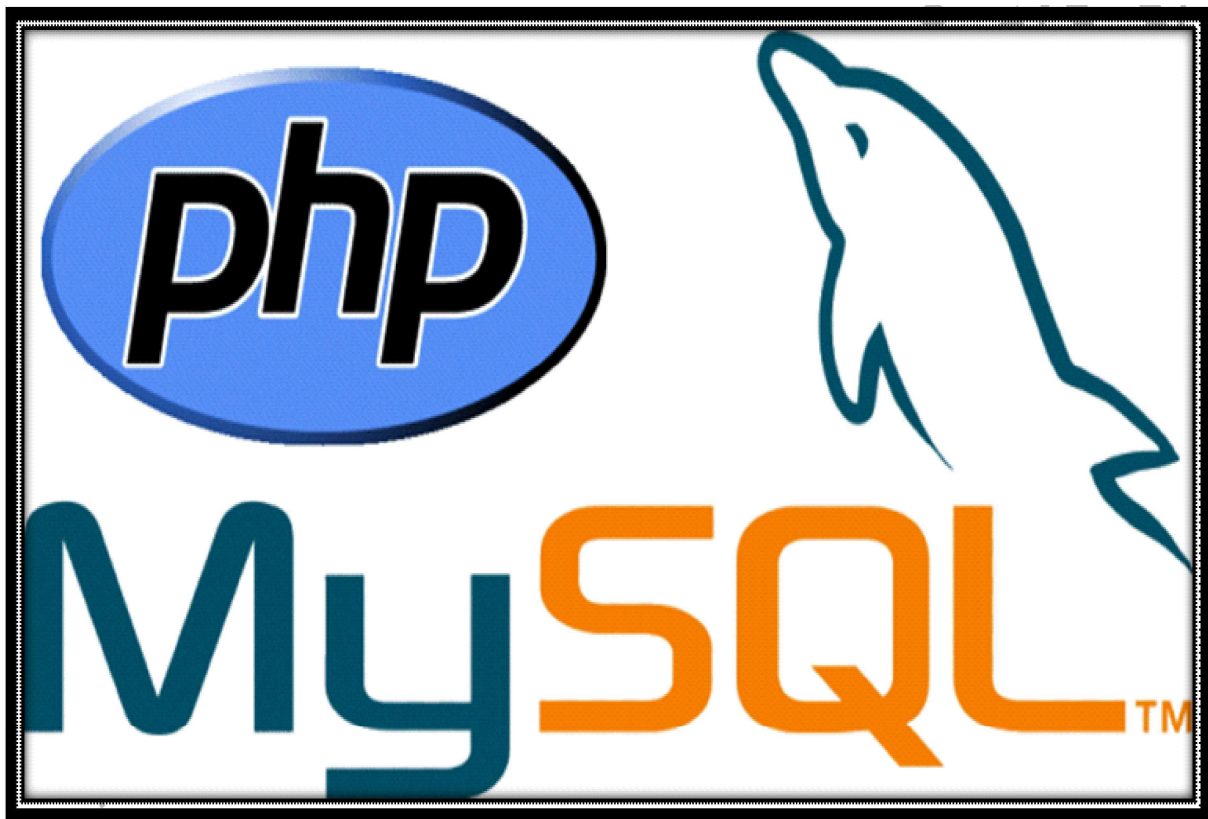

B.SC (MPCS) -IV SEM

PHP & MYSQL



PHP and My SQL SYLLABAS

Unit-I

Building blocks of PHP: Variables, Data Types, Operators and Expressions, Constants.

Flow Control Functions in PHP: Switching Flow, Loops, Code Blocks and Browser Output.

Working with Functions: Defining Functions, Calling functions, returning the values from User- Defined Functions, Variable Scope, Saving State between Function calls with the Static statement, more about arguments.

Unit-II

Working with Arrays: Arrays, Creating Arrays, Some Array-Related Functions.

Working with Objects: Creating Objects, Object Instance.

Working with Strings, Dates and Time: Formatting Strings with PHP, Investigating Strings with PHP, Manipulating Strings with PHP, Using Date and Time Functions in PHP.

Unit-III

Working with Forms: Creating Forms, Accessing Form - Input with User defined Arrays, Combining HTML and PHP code on a single Page, Using Hidden Fields to save state, Redirecting the user, Sending Mail on Form Submission, Working with File Uploads. Working with Cookies and User Sessions: Introducing Cookies, Setting a Cookie with PHP, Session Function Overview, Starting a Session, Working with session variables, passing session IDs in the Query String, Destroying Sessions and Unsetting Variables, Using Sessions in an Environment with Registered Users.

Unit-IV

Working with Files and Directories: Including Files with include(), Validating Files, Creating and Deleting Files, Opening a File for Writing, Reading or Appending, Reading from Files, Writing or Appending to a File, Working with Directories, Open Pipes to and from Process Using popen (), Running Commands with exec(), Running Commands with system () or passthru ().

Working with Images: Understanding the Image-Creation Process, Necessary Modifications to PHP, Drawing a New Image, Getting Fancy with Pie Charts, Modifying Existing Images, Image Creation from User Input.

Unit-V

Interacting with MySQL using PHP: MySQL Versus MySQLi Functions, Connecting to MySQL with PHP, Working with MySQL Data. Creating an Online Address Book: Planning and Creating Database Tables, Creating Menu, Creating Record Addition Mechanism, Viewing Records, Creating the Record Deletion Mechanism, Adding Sub-entities to a Record.

References:

1. Julie C. Meloni, PHP MySQL and Apache, SAMS Teach Yourself, Pearson Education (2007).
2. Xue Bai Michael Ekedahl, The Web Warrior Guide to Web Programming, Thomson (2006).

Building blocks of PHP

Unit - I :: Flow Control Functions in PHP

Working with Functions

Q1) Explain about PHP? Explain its features.

PHP is a server side scripting language, and it is a powerful tool for making dynamic and interactive Web pages. PHP is a widely-used, free, and efficient alternative to competitors such as Microsoft's ASP.

- PHP is a server side scripting language.
- PHP is an interpreted language, i.e. there is no need for compilation.
- PHP is an object-oriented language.
- PHP is an open-source scripting language.
- PHP is simple and easy to learn language.

PHP Features

Performance: Script written in PHP executes much faster than those scripts written in other languages such as JSP & ASP.

Open Source Software: PHP source code is free available on the web, you can develop all the version of PHP according to your requirement without paying any cost.

Platform Independent: PHP are available for WINDOWS, MAC, LINUX & UNIX operating system. A PHP application developed in one OS can be easily executed in other OS also.

Compatibility: PHP is compatible with almost all local servers used today like Apache, IIS etc.

Embedded: PHP code can be easily embedded within HTML tags and script.

What is a PHP File?

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

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

PHP files have extension ".php"

What Can PHP Do?

- PHP can generate dynamic page content
- PHP can create, open, read, write, delete, and close files on the server
- PHP can collect form data
- PHP can send and receive cookies
- PHP can add, delete, modify data in your database
- PHP can be used to control user-access
- PHP can encrypt data

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

Why PHP?

- PHP runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
- PHP is compatible with almost all servers used today (Apache, IIS, etc.)
- PHP supports a wide range of databases
- PHP is free. Download it from the official PHP resource: www.php.net
- PHP is easy to learn and runs efficiently on the server side

Q2) Explain about Building blocks of PHP:-

PHP language contains several basic concepts.

- **Variables**
- **Data types**
- **Operators**
- **Expressions**
- **Constants.**

PHP variables:

- Variables are “containers” for storing information.
- A PHP variable is a named memory location that holds data.
- A variable is a temporary storage that is used to store data temporarily.
- In PHP, a variable is declared using \$ sign followed by variable name.

Rules for PHP variables:

- A variable starts with the \$ sign, followed by the name of the variable
- A variable name must start with a letter or the underscore character
- A variable name cannot start with a number
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- Variable names are case-sensitive (\$age and \$AGE are two different variables)

Output Variables:-

- The PHP echo statement is often used to output data to the screen.

The following example will show how to output text and a variable:

Syntax of declaring a variable in PHP is given below:

```
$variablename=value;
<?php
$str="hello string";
$x=200;
$y=44.6;
echo "string is: $str <br/>";
echo "integer is: $x <br/>";
echo "float is: $y <br/>";
?>
```

Output:

string is: hello string

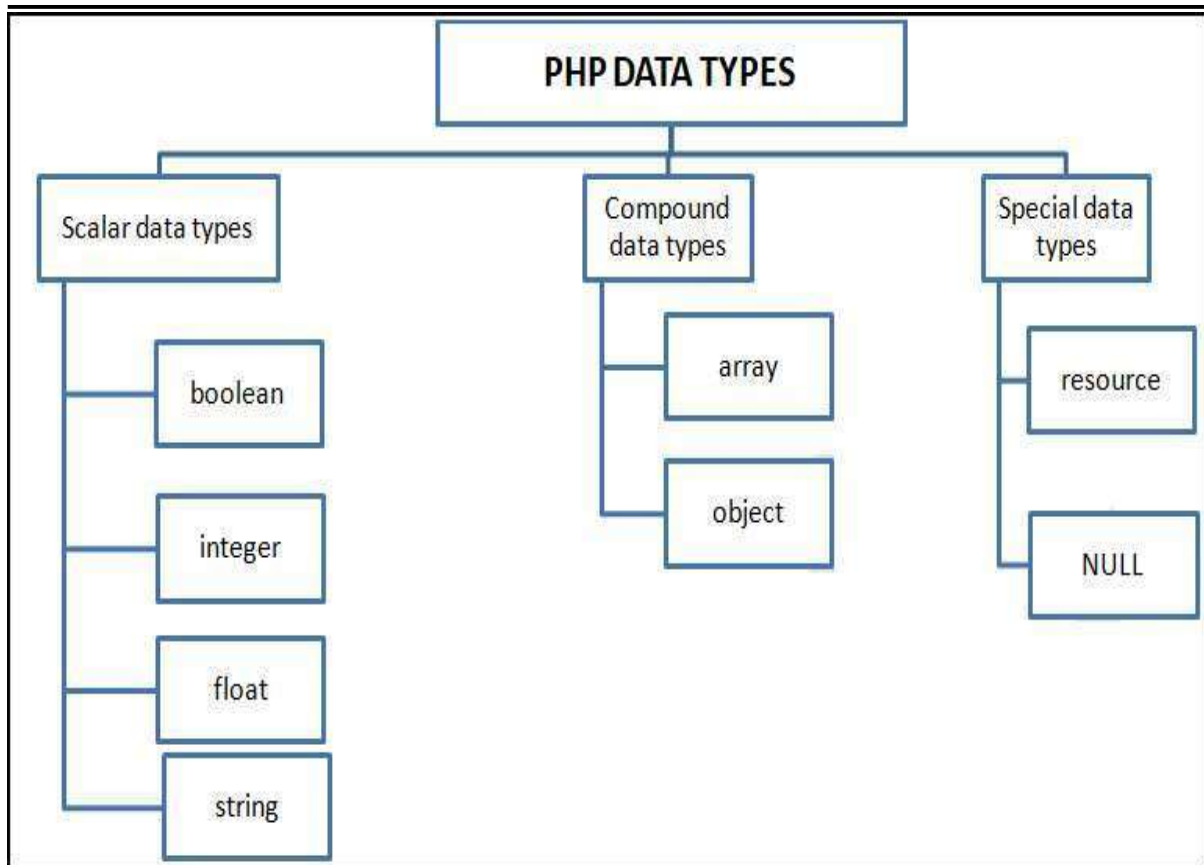
integer is: 200

float is: 44.6

PHP Data Types:

PHP supports different types of data types. Those are

- ☐ Scalar data types
- ☐ compound data types
- ☐ special data types



- A string is a sequence of characters, like "Hello world!".
- A string can be any text inside quotes. You can use single or double quotes:
Ex:-

`$x = "Hello world!";`

`$y = 'Hello world!';`

Integer

- An integer data type is a non-decimal number.

Rules for integers:

- An integer must have at least one digit
- An integer must not have a decimal point
- An integer can be either positive or negative

Ex:- `$x=58;`

Float

A float (floating point number) is a number with a decimal point.

Ex:- `$x=10.35;`

Boolean

A Boolean represents two possible states: TRUE or FALSE.

Ex:- `$x = true;`

`$y = false;`

Booleans are often used in conditional testing. You will learn more about conditional testing in a later chapter of this tutorial.

Array

- An array stores multiple values in one single variable.
- In the following example \$cars is an array.

Ex:-

```
$cars = array("Volvo","BMW","Toyota");
```

Object

- An object is a data type which stores data and information and it can process that data.
- In PHP, an object must be explicitly declared.

NULL Value

- Null is a special data type which can have only one value: NULL.
- A variable of data type NULL is a variable that has no value assigned to it.
- Tip: If a variable is created without a value, it is automatically assigned a value of NULL.
- Variables can also be emptied by setting the value to NULL:

Ex:-

```
$x = null;
```

Resource

- Resource is a special data type is not an actual data type.
- It is the storing of a reference to functions and resources external to PHP.

PHP Operators:

PHP Operator is a symbol. Operators are used to perform operations on variables and values. PHP divides the operators in the following groups:

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

PHP Arithmetic Operators

The PHP arithmetic operators are used with numeric values to perform common arithmetical operations, such as addition, subtraction, multiplication etc.

Operator	Name	Example
+	Addition	$\$x + \y
-	Subtraction	$\$x - \y
*	Multiplication	$\$x * \y
/	Division	$\$x / \y
%	Modulus	$\$x \% \y
**	Exponentiation	$\$x ** \y

PHP Assignment Operators

The PHP assignment operators are used with numeric values to write a value to a variable. The basic assignment operator in PHP is “=”. It means that the left operand gets set to the value of the assignment expression on the right.

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

PHP Comparison Operators

The PHP comparison operators are used to compare two values (number or string):

Operator	Name	Example
<code>==</code>	Equal	<code>\$x == \$y</code>
<code>!=</code>	Not equal	<code>\$x != \$y</code>
<code><></code>	Not equal	<code>\$x < > \$y</code>
<code>></code>	Greater than	<code>\$x > \$y</code>
<code><</code>	Less than	<code>\$x < \$y</code>
<code>>=</code>	Greater than or equal to	<code>\$x >= \$y</code>
<code><=</code>	Less than or equal to	<code>\$x <= \$y</code>

PHP Increment / Decrement Operators

The PHP increment operators are used to increment a variable's value.

The PHP decrement operators are used to decrement a variable's value.

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

PHP Logical Operators

The PHP logical operators are used to combine conditional statements.

Operator	Name	Example
<code>and</code>	And	<code>\$x and \$y</code>
<code>or</code>	Or	<code>\$x or \$y</code>
<code>&&</code>	And	<code>\$x && \$y</code>
<code> </code>	Or	<code>\$x \$y</code>
<code>!</code>	Not	<code>!\$x</code>

The Concatenation Operator

The concatenation operator is represented by a single period (.). Treating both operands as strings, this operator appends the right-side operand to the left-side operand.

Ex:-

`"hello"." world"`

Output:-

`"hello world"`

Expressions:-

In programming, an expression is any legal combination of symbols that represents a value. Each programming language and application has its own rules. Every expression consists of at least one operand and can have one or more operators. Operands are values, whereas operators are symbols that represent particular actions.

In the expression :

$x + 5$

x and 5 are operands, and + is an operator.

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.

Syntax:

define(name, value, case-insensitive);

Parameters:

name: Specifies the name of the constant

value: Specifies the value of the constant

case-insensitive: Specifies whether the constant name should be case-insensitive.

Default is false

ex:-

```
define("GREETING", "Welcome to SHDC PHP CLASS!", true);
```

Q3) Explain about Flow(conditional) control statements in PHP:-

Normally PHP execute the script in line by line. Sometimes developer want to execute the script based on the particular condition. Then we use flow control functions.

- **If –Else**
- **Switch**
- **For**
- **While**
- **Do-while**

If-Else:-

PHP if else statement is used to test condition. There are various ways to use if statement in PHP.

- if
- if-else
- if-else-if
- nested if

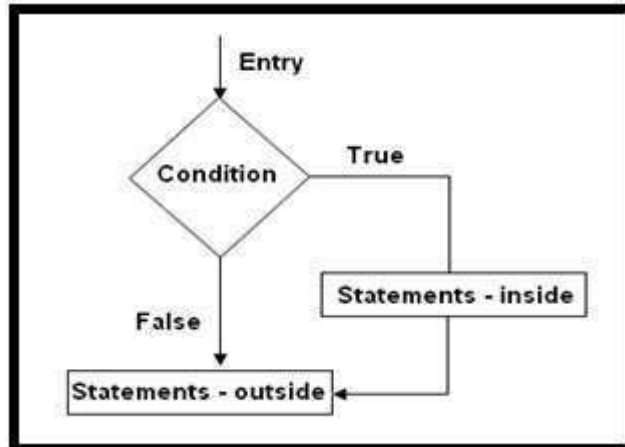
PHP If Statement

PHP if statement is executed if condition is true.

Syntax:-

```
if(condition)
{
    //code to be executed
}
```

Flowchart:-



Example:-

```
<?php
$num=12;
if($num<100)
{
    echo "$num is less than 100";
}
echo "Bye";
?>
```

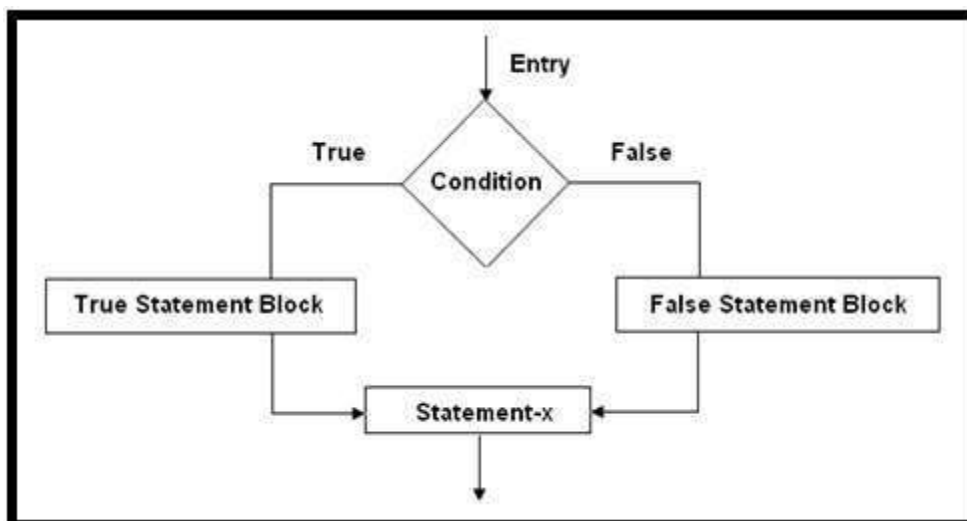
PHP If-else Statement:-

PHP if-else statement is executed whether condition is true or false.

Syntax:-

```
if(condition)
{
    //code to be executed if true
}
else
{
    //code to be executed if false
}
```

Flowchart:-



Example:-

```
<?php
$num=12;
if($num%2==0)
{
    echo "$num is even number";
}
else{
    echo "$num is odd number";
}
?>
```

PHP Switch:-

PHP switch statement is multi way decision making statement.

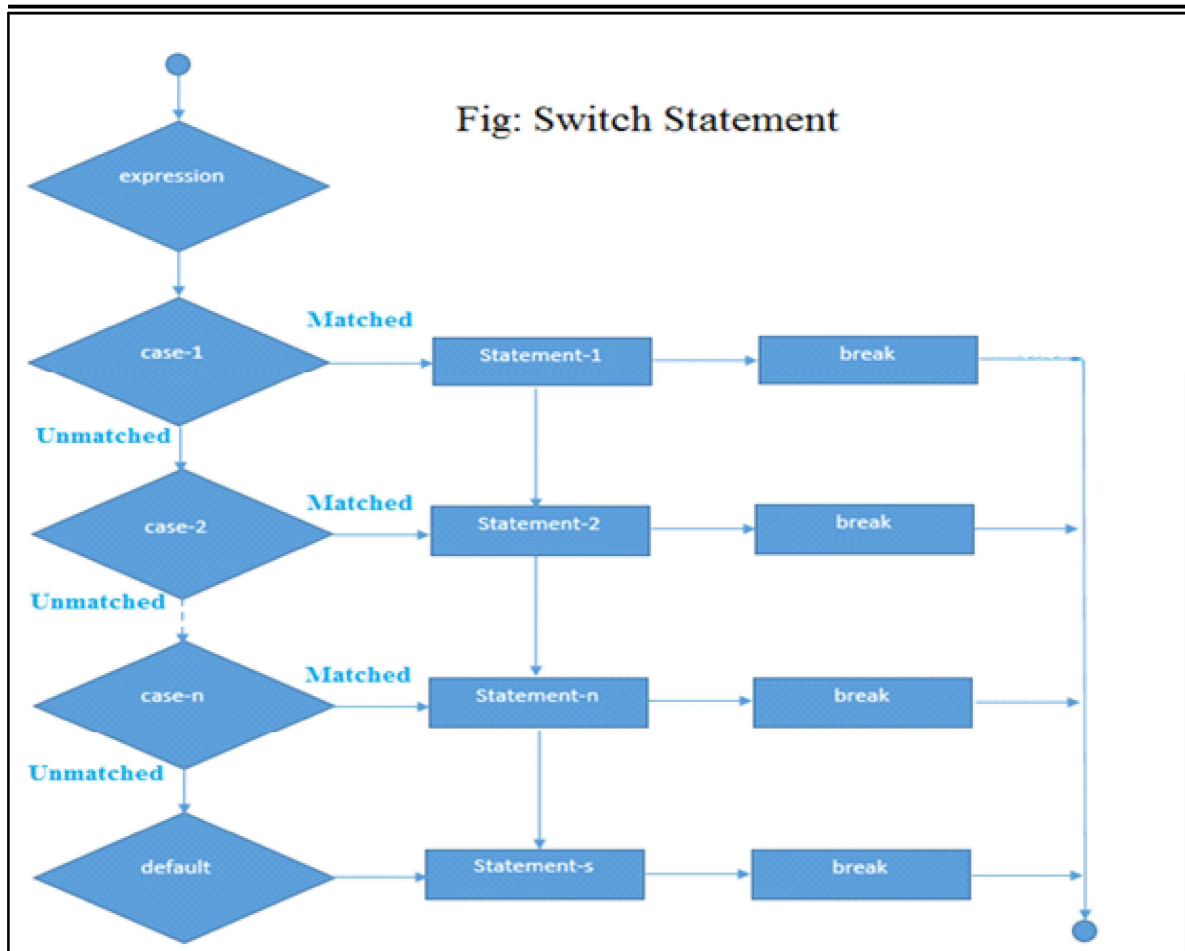
This statement is used when the program contains many number of conditions like if-else-if.

The general form of switch statement is

Syntax:-

```
switch(expression)
{
    case value1:
        //code to be executed
        break;
    .....
    default:
        code to be executed if all cases are not matched;
}
```

PHP Switch Flowchart:-



PHP Switch Example:-

```

<?php
$num=20;
switch($num)
{
case 10:
echo("number is equals to 10");
break;
case 20:
echo("number is equal to 20");
break;
case 30:
echo("number is equal to 30");
break;
default:
echo("number is not equal to 10, 20 or 30");
}
?>

```

Q4) Explain about Php loops:-

Looping statements allow the set of instructions to be executed repeatedly until the condition is reached. And loops are used to repeat a block of code. and These looping statements contain 3 control statements

1. Variable initialization
2. Condition or expression
3. Increment or decrement operator

Various looping statements are:

- **While loop**
- **Do – while loop**
- **For loop**

While Loop:-

While loop is an entry control loop.

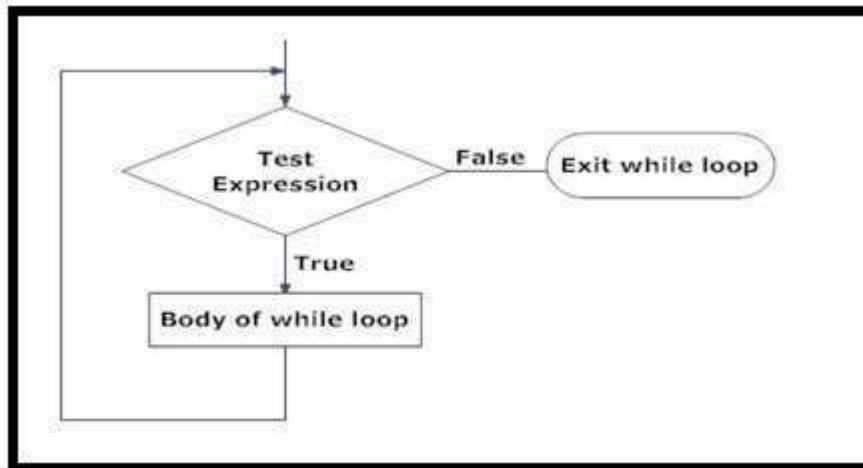
Condition is checked and if it is true, then group of statements or body of loop is executed.

It will execute again and again till condition becomes false.

Syntax -

```
while (test condition)
{
    Block of statements;
    Increment/decrement operator;
}
Statement -x;
```

Flow Chart:-



Example-

```
<?php
$n=1;
while($n<=10)
{
    echo "$n<br/>";
    $n++;
}
?>
```

Do – while Loop:-

Do - While is an exit control loop. First body of the loop is executed and then the condition is checked. If condition is true, then body of the loop is executed otherwise it will exit from loop.

Syntax -

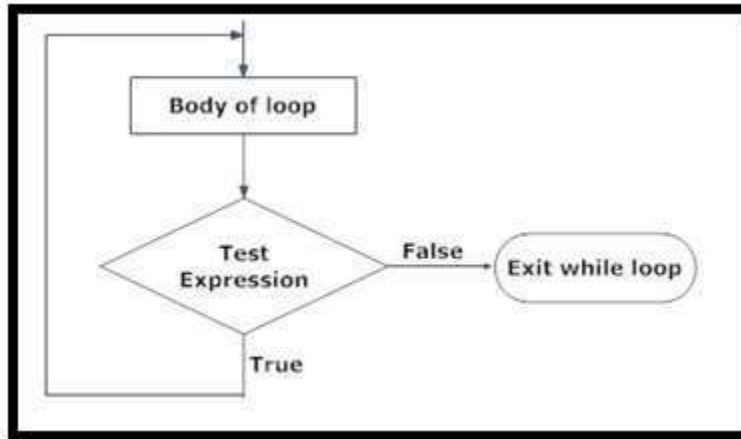
```
do
{
    Block of statements;

    Increment/decrement operator;

} while (test condition);

Statement -x;
```

Flow Chart :-



Example:

```
<?php
$n=1;
do
{
echo "$n<br/>";
$n++;
}
while($n<=10);
?>
```

For Loop:-

When a user want to execute set of statements repeatedly until the specified condition is false then we use for loops.

Syntax -

```
for(initialization ; test expression; increment/decrement)
```

```
{
```

```
    Block of loop;
```

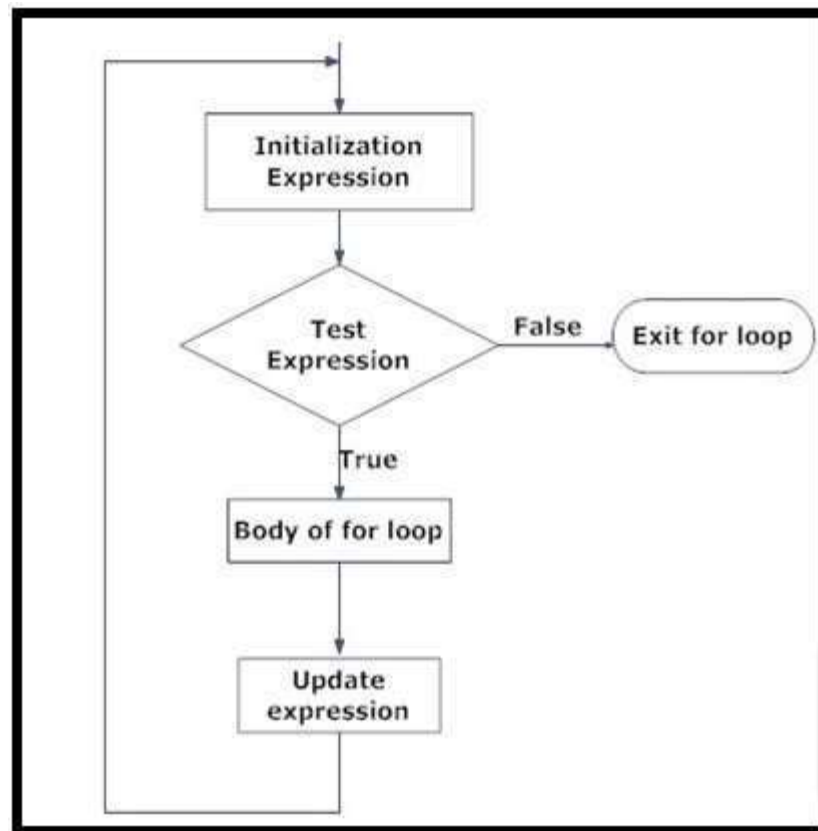
```
}
```

```
Statement -x;
```

The initial expression is at the beginning of for loop. Then, the test expression is checked by the program. If the test expression is false, for loop is terminated. But, if test expression is true then, the statements are executed and update expression is updated.

Again, the test expression is checked. If it is false, loop is terminated and if it is true, the same process repeats until test expression is false.

Flow Chart: -



Example:-

```
<?php
for($n=1;$n<=10;$n++)
{
    echo "$n<br/>";
}
?>
```

Q5) Explain about Code blocks and browser outputs:-

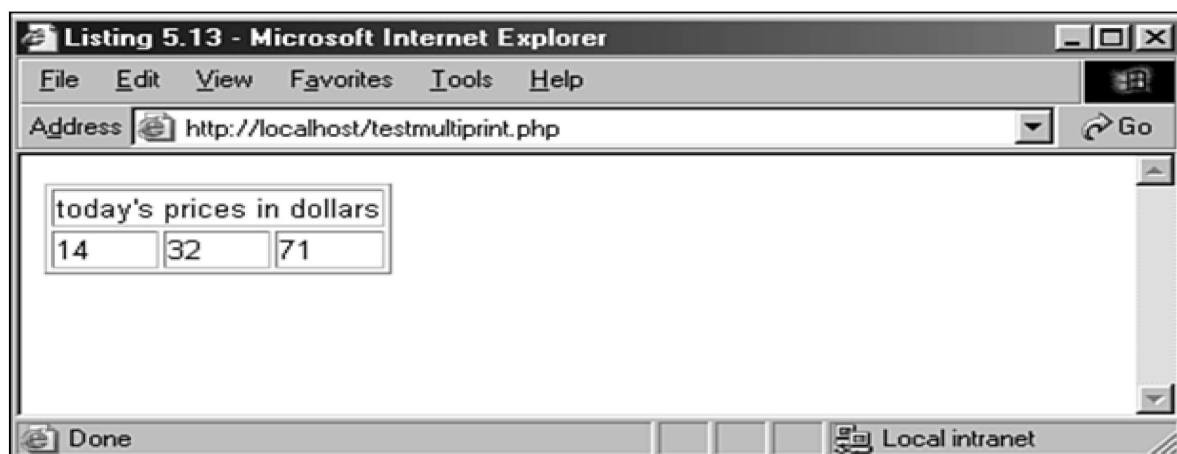
Imagine a script that outputs a table of values only when a variable is set to the Boolean value true. PART-I shows a simplified HTML table constructed with the code block of an if statement.

PART-I is a Code Block Containing Multiple print() Statements.

Part-I	Part-II
<pre><?php \$display_prices = true; if (\$display_prices) { print "<table border=\"1\">"; print "<tr><td colspan=\"3\">"; print "today's prices in dollars"; print "</td></tr>"; print "<tr><td>14</td><td>32</td><td>71</td></tr>"; print "</table>"; } ?></pre>	<pre><?php \$display_prices = true; if (\$display_prices) { ?> <table border="1"> <tr><td colspan="3">today's prices in dollars</td></tr> <tr><td>14</td><td>32</td><td>71</td> </table> <?php } ?></pre>

If \$display_prices is set to true in part -I line 2, the table is printed. For the sake of readability, we split the output into multiple print() statements, and once again escape any quotation marks.

Put these lines into a text file called testmultiprint.php, and place this file in your Web server document root. When you access this script through your Web browser, it should look like bellow Figure



There's nothing wrong with the way this is coded, but we can save ourselves some typing by simply slipping back into HTML mode within the code block.

In the PART-II the important thing to note here is that the shift to HTML mode on line 5 only occurs if the condition of the if statement is fulfilled. This can save us the bother of escaping quotation marks and wrapping our output in print() statements. It might, however, affect the readability of our code in the long run, especially as our script grows larger.

Q6) Explain about Functions in PHP:-

PHP Functions:-

PHP function is a piece of code that can be reused many times. It can take input as argument list and return value. There are thousands of built-in functions in PHP. In PHP, we can define Conditional function, Function within Function and Recursive function also.

PHP User Defined Functions

- Besides the built-in PHP functions, we can create our own functions.
- 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.

•User Defined Function in PHP:-

A user defined function declaration starts with the word "function":

Syntax:-

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

Example:-

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

PHP Function Arguments

We can pass the information in PHP function through arguments which is separated by comma. PHP supports Call by Value (default), Call by Reference, Default argument values and Variable-length argument list.

Example:-

```
<?php
function sayHello($name)
{
    echo "Hello $name<br/>";
}
sayHello("Sonoo");
sayHello("Vimal");
sayHello("John");
?>
```

PHP Function: Default Argument Value

We can specify a default argument value in function. While calling PHP function if you don't specify any argument, it will take the default argument. Let's see a simple example of using default argument value in PHP function

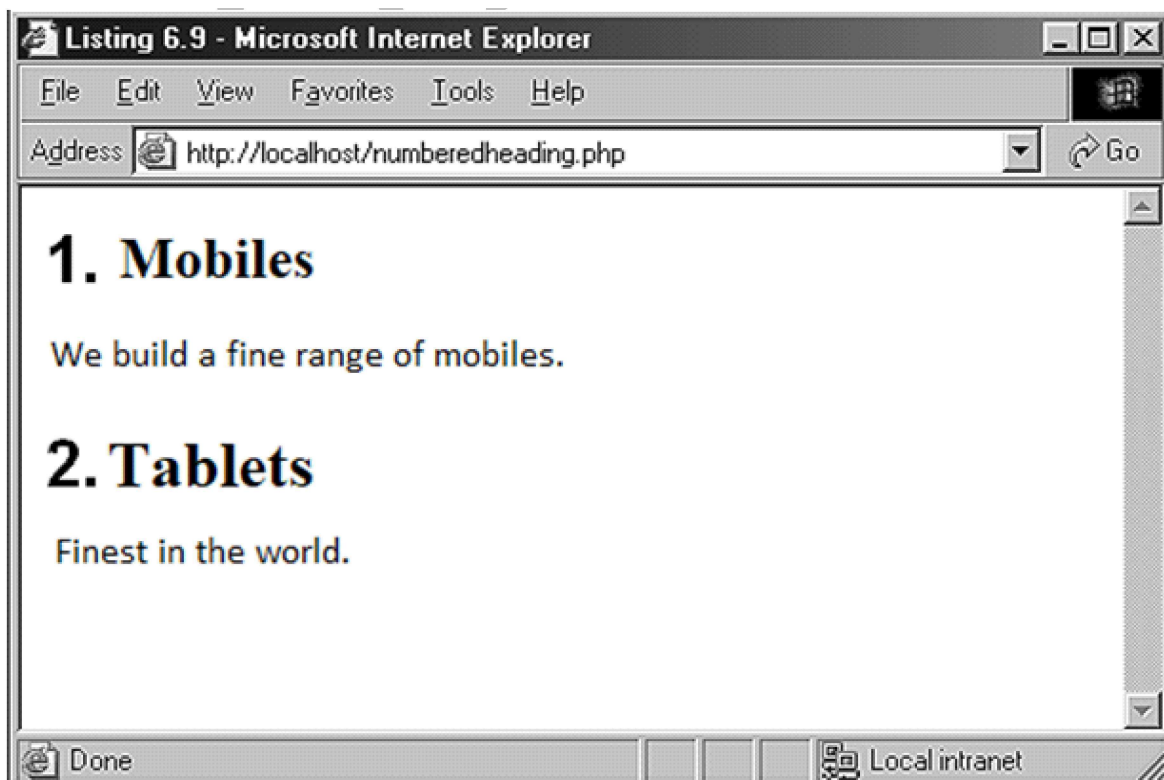
```
<?php
function sayHello($name="Sonoo")
{
    echo "Hello $name<br/>";
}
sayHello("Rajesh");
sayHello();//passing no value
sayHello("John");
?>
```

Q7) Explain about Saving State Between Function Calls with the static Statement?

Local variables within functions have a short life. After executing the function local variables are died.

Let's assume that we want a function to keep track of the number of times it has been called so that numbered headings can be created by a script. We could, of course, use the global statement to do this

```
<?php
$num_of_calls = 0;
function numberedHeading($txt)
{
    global $num_of_calls;
    $num_of_calls++;
    echo "<h1>".$num_of_calls." ".$txt."</h1>";
}
numberedHeading("mobiles");
echo "<p>We build a fine range of tablets.</p>";
numberedHeading("tablets");
echo "<p>Finest in the world.</p>";
?>
```



Using the static Statement to Remember the Value of a Variable Between Function Calls:-

```
<?php
function numberedHeading($txt)
{
    static $num_of_calls = 0;
    $num_of_calls++;
    echo "<h1>".$num_of_calls." ". $txt."</h1>";
}
numberedHeading("mobiles");
echo "<p>We build a fine range of mobiles.</p>";
numberedHeading("tablets");
echo "<p>Finest in the World.</p>";
?>
```

Q8) Explain about PHP Variables Scope (types of variables):-

In PHP, variables can be declared anywhere in the script.

The scope of a variable is the part of the script where the variable can be referenced/used.

PHP has three different variable scopes:

- **local**
- **global**
- **static**

Global and Local Scope:-

A variable declared inside a function has a LOCAL SCOPE and can only be accessed INSIDE a function:

Example:-

```
<?php
$x = 5; // global scope

function myTest() {
    // using x inside this function will generate an error
    echo "<p>Variable x inside function is: $x</p>";
}
myTest();
echo "<p>Variable x outside function is: $x</p>";
?>
```

A variable declared within a function has a LOCAL SCOPE and can only be accessed within that function:

Example:-

```
<?php
function myTest() {
    $x = 5; // local scope
    echo "<p>Variable x inside function is: $x</p>";
}
myTest();

// using x outside the function will generate an error
echo "<p>Variable x outside function is: $x</p>";
?>
```

You can have local variables with the same name in different functions, because local variables are only recognized by the function in which they are declared.

PHP The global Keyword:-

The global keyword is used to access a global variable from within a function.

To do this, use the global keyword before the variables (inside the function):

Example:-

```
<?php
$x = 5;
$y = 10;
```

```
function myTest()
{
    global $x, $y;
    $y = $x + $y;
}
myTest();
echo $y; // outputs 15
?>
```

PHP also stores all global variables in an array called `$GLOBALS[index]`. The index holds the name of the variable. This array is also accessible from within functions and can be used to update global variables directly.

Example:-

```
<?php
$x = 5;
$y = 10;

function myTest()
{
    $GLOBALS['y'] = $GLOBALS['x'] + $GLOBALS['y'];
}
myTest();
echo "$y"; // outputs 15
?>
```

PHP The static Keyword:-

Normally, when a function is completed/executed, all of its variables are deleted. However, sometimes we want a local variable NOT to be deleted. We need it for a further job.

To do this, use the static keyword when you first declare the variable:

Example:-

```
<?php
function myTest() {
    static $x = 0;
    echo "$x    ";
    $x++;
}
```

```
myTest();  
myTest();  
myTest();  
?>
```

Unit-II : working with arrays

Q1) What is an Array? Explain about types of arrays in PHP?

An array is a special variable, which can hold more than one value at a time. If you have a list of items (a list of car names, for example), storing the cars in single variables could look like this:

```
$cars1 = "Volvo";  
$cars2 = "BMW";  
$cars3 = "Toyota";
```

An array can hold many values under a single name, and you can access the values by referring to an index number.

Create an Array in PHP:

In PHP, the array() function is used to create an array:

```
array();
```

In PHP, 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

a)Indexed Arrays:-

PHP indexed array is an array which is represented by an index number by default. All elements of array are represented by an index number which starts from 0. The index can be assigned automatically (index always starts at 0), like this:

Program:-

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

b)Associative array:-

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

To store the salaries of employees in an array, a numerically indexed array would not be the best choice. Instead, we could use the employees names as the keys in our associative array, and the value would be their respective salary.

NOTE :-

" Don't keep associative array inside double quote while printing otherwise it would not return any value.

Program:

```
<?php
/* First method to associate create array. */
$salaries = array("mani" => 2000, "raju" => 1000, "krishna" => 500);

echo "Salary of mani is ". $salaries['mani'] . "<br />";
echo "Salary of raju is ". $salaries['raju']. "<br />";
echo "Salary of krishna is ". $salaries['krishna']. "<br />";

/* Second method to create array. */
$salaries['mani'] = "high";
$salaries['raju'] = "medium";
$salaries['krishna'] = "low";

echo "Salary of mani is ". $salaries['mani'] . "<br />";
echo "Salary of raju is ". $salaries['raju']. "<br />";
echo "Salary of krishna is ". $salaries['krishna']. "<br />";
?>
```

c)Multidimensional Arrays

PHP multidimensional array is also known as array of arrays. It allows you to store tabular data in an array. PHP multidimensional array can be represented in the form of matrix which is represented by **row * column**.

Example

```
$emp = array
(
    array(1,"sonoo",400000),
    array(2,"john",500000),
    array(3,"rahul",300000)
);
```

Program:-

```
<?php
    $emp = array
    (
        array(1,"sonoo",400000),
        array(2,"john",500000),
        array(3,"rahul",300000)
    );

    for ($row = 0; $row < 3; $row++) {
        for ($col = 0; $col < 3; $col++) {
            echo $emp[$row][$col]. " ";
        }
        echo "<br/>";
    }
?>
```

Q2) Explain about different types ARRAY functions:-

a)PHP array() function:-

PHP array() function creates and returns an array. It allows you to create

- indexed,
- associative and
- multidimensional arrays.

Example:-

```
<?php
$names=array("krishna","raju","ravi","murali");
echo "Names are: $names[0], $names[1], $names[2] and $names[3]";
?>
```

b)PHP array_change_key_case() function

PHP array_change_key_case() function changes the case of all key of an array.

Note: It changes case of key only.

Example:-

```
<?php
$salary=array("Sonoo"=>"550000","Vimal"=>"250000","Ratan"=>"200000");
print_r(array_change_key_case($salary,CASE_UPPER));
?>
```

c)PHP count() function

PHP count() function counts all elements in an array.

Example:-

```
<?php
$names=array("krishna","khan","kasi","karuna");
echo count($names);
?>
```

d)PHP sort() function

PHP sort() function sorts all the elements in an array.

Example:

```
<?php
$names=array("krishna","khan","kasi","karuna");
sort($names);
foreach( $names as $s )
{
    echo "$s<br />";
}
?>
```

e)PHP array_reverse() function

PHP array_reverse() function returns an array containing elements in reversed order.

Example:-

```
<?php
$names=array("krishna","raju","ravi","murali");
$reversenames=array_reverse($names);
foreach( $reversenames as $s )
{
    echo "$s<br />";
}
?>
```

f)PHP array_search() function

PHP array_search() function searches the specified value in an array.

It returns key if search is successful.

Example:-

```
<?php
$names=array("krishna","raju","ravi","murali");
$key=array_search("ravi",$names);
echo $key;
?>
```

g)PHP array_sum() Function

The array_sum() function returns the sum of all the values in the array.

Example:-

```
<?php
$a=array(5,15,25);
echo array_sum($a);
?>
```

Q3) Explain about how to create an Object in PHP:- **Object in Java**

An entity that has state and behaviour is known as an object .

e.g. chair, bike, marker, pen, table, car etc. It can be physical or logical.

An object has three characteristics:

- state:** represents data (value) of an object.
- behavior:** represents the behavior (functionality) of an object such as deposit, withdraw etc.
- identity:** Object identity is typically implemented via a unique ID. The value of the ID is not visible to the external user. But, it is used internally by the JVM to identify each object uniquely.

For Example: Pen is an object. Its name is Reynolds, color is white etc. known as its state. It is used to write, so writing is its behavior.

Object is an instance of a class. Class is a template or blueprint from which objects are created. So object is the instance (result) of a class.

Object Definitions:

- Object is a real world entity.
- Object is a run time entity.
- Object is an entity which has state and behavior.
- Object is an instance of a class.

Class:

Class is a user-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.

In PHP declare a class using the class keyword, followed by the name of the class and a set of curly braces ({}).

Syntax Create Class in PHP:-

```
<?php
class MyClass
{
    // Class properties and methods go here
}
?>
```

After creating the class, a new class can be instantiated and stored in a variable using the

new keyword:

Syntax:-

\$obj = new MyClass;

In PHP, to see the contents of the class, use var_dump(). The var_dump() function is used to display structured information (type and value) about one or more variables.

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.

Syntax Class in PHP

```
<?php
class phpClass
{
    var $var1;
    var $var2 = "constant string";
    function myfunc ($arg1, $arg2)
    {
        [..]
    }
    [..]
}
```

```
}  
?>
```

Syntax Create Object in PHP:-

```
<?php  
class MyClass  
{  
    // Class properties and methods go here  
}  
$obj = new MyClass;  
var_dump($obj);  
?>
```

Q4) Explain about Strings and its functions:-

String in PHP:-

string is a sequence of characters i.e. used to store and manipulate text. There are 4 ways to specify string in PHP.

Single quoted

Double quoted

Single Quoted PHP String

It is simple and easy way to specify string in php. We can create a string in PHP by enclosing text in a single quote.

Use String in PHP

```
<?php  
$str='Hello php I am single quote String';  
echo $str;  
?>
```

Output:-

Hello php I am single quote String

Double Quoted PHP String:-

In PHP, we can also specify string through enclosing text within double quote

Use String in PHP

```
<?php
$str="Hello php I am double quote String";
echo $str;
?>
```

Output:

Hello php I am double quote String

String functions:-

PHP have lots of predefined function which is used to perform operation with string some functions are strlen(), strrev(), strpos() etc.

PHP strlen() function

strlen() function returns the length of a string. In below example strlen() function is used to return length of "Hello world!"

strlen() example in php

```
<?php

echo strlen("Hello world!");

?>
```

Output

12

PHP str_word_count() function

str_word_count() function is used to count numbers of words in given string.

str_word_count() example in php

```
<?php

echo str_word_count("Hello world!");

?>
```

Output

2

PHP strrev() function

strrev() function is used to reverse any string.

strrev() example in php

```
<?php

echo strrev("Hello world!");

?>
```

Output

!dlrow olleH

PHP strpos() function

strpos() function is used to search specific position of any words in given string.

strpos() example in php

```
<?php

echo strpos("Hello world!", "world");

?>
```

Output

6

PHP str_replace() function

str_replace function is used to replace some characters with some other characters in a string. In below example we replace world with Faiz.

str_replace() example in php

```
<?php

echo str_replace("world", "Faiz", "Hello world!");

?>
```

Output

Hello Faiz!

PHP strtolower() function

strtolower() function is used to convert uppercase letters into lowercase letters.

strtolower() example in php

```
<?php

$str="Hello friends i am HITESH";
$str=strtolower($str);
echo $str;

?>
```

Output

hello friends i am hitesh

PHP strtoupper() function

PHP strtoupper() function is used to convert lowercase latter into uppercase latter.

strtoupper() example in php

```
<?php

$str="Hello friends i AM Hitesh";
$str=strtoupper($str);
echo $str;

?>
```

Output

HELLO FRIENDS I AM HITESH

Q5) Explain baout Date and time functions:-

The date/time functions allow you to get the date and time from the server where your PHP script runs. You can then use the date/time functions to format the date and time in several ways.

The PHP Date() Function:-

- The PHP date() function convert a timestamp to a more readable date and time.
- The computer stores dates and times in a format called UNIX Timestamp, which measures time as a number of seconds since the beginning of the Unix time format.

Ex:-

```
<?php

$today = date("d/m/Y");
echo $today;

?>
```

Formatting the Dates and Times with PHP:-

Here are some the date-related formatting characters that are commonly used in format string:

- d - Represent day of the month; two digits with leading zeros (01 or 31)
- D - Represent day of the week in text as an abbreviation (Mon to Sun)
- m - Represent month in numbers with leading zeros (01 or 12)
- M - Represent month in text, abbreviated (Jan to Dec)
- y - Represent year in two digits (08 or 14)
- Y - Represent year in four digits (2008 or 2014)

Ex:-

```
<?php
echo date("d/m/Y") . "<br>";
echo date("d-m-Y") . "<br>";
echo date("d.m.Y");
?>
```

The PHP time() Function:-

The time() function is used to get the current time as a Unix timestamp (the number of seconds since the beginning of the Unix epoch: January 1 1970 00:00:00 GMT).

Ex:-

```
<?php
// Executed at March 05, 2014 07:19:18
$timestamp = time();
echo($timestamp);
?>
```

The above example produce the following output.

1394003958

We can convert this timestamp to a human readable date through passing it to the previously introduce date() function.

Example:-

```
<?php
$timestamp = 1394003958;
echo(date("F d, Y h:i:s", $timestamp));
?>
```

The PHP mktime() Function:-

The mktime() function is used to create the timestamp based on a specific date and time. If no date and time is provided, the timestamp for the current date and time is returned.

The syntax of the mktime() function can be given with:

mktime(hour, minute, second, month, day, year)

The following example displays the timestamp corresponding to 3:20:12 pm on May 10, 2014:

Example:-

```
<?php
// Create the timestamp for a particular date
echo mktime(15, 20, 12, 5, 10, 2014);
?>
```

End of unit-II

Unit-III : Forms, Files uploads, Sending Mail

Q1)What is a Form? How to create a form?

A Document that containing blank fields, that the user can fill the data or user can select the data. Casually the data will store in the data base

PHP Post Form:-

Post request is widely used to submit form that have large amount of data such as file upload, image upload, login form, registration form etc.

The data passed through post request is not visible on the URL browser so it is secured. You can send large amount of data through post request.

Let's see a simple example to receive data from post request in PHP.

File: form1.html

```
<html>
<body>
<form action="login.php" method="post">
Name: <input type="text" name="name"><br>
E-mail: <input type="text" name="email"><br>
<input type="submit">
</form>
</body>
</html>
File: login.php
<?php
$name=$_POST["name");//receiving name field value in $name variable
$email=$_POST["email");//receiving password field value in $password variable
echo "Welcome: $name<br/> your E-mail Id is: $email";
?>
```

Q2) Explain about Mail function. Sending mail on form submission:-

PHP Mail:-

PHP mail() function is used to send email in PHP. You can send text message, html message and attachment with message using PHP mail() function.

PHP mail() function Syntax:-

```
mail (string $to, string $subject, string $message [, string $additional_headers [, string $additional_parameters ]]);
```

\$to: specifies receiver or receivers of the mail. The receiver must be specified one of the following forms.

user@example.com

user@example.com, anotheruser@example.com

User <user@example.com>

User <user@example.com>, Another User <anotheruser@example.com>

\$subject: represents subject of the mail.

\$message: represents message of the mail to be sent.

Note: Each line of the message should be separated with a CRLF (\r\n) and lines should not be larger than 70 characters.

\$additional_headers (optional):

specifies the additional headers such as From, CC, BCC etc. Extra additional headers should also be separated with CRLF (\r\n).

Example:- File: mailer.php

```
<?php
ini_set("sendmail_from", "nimu725@gmail.com");
$to = "nimu822@gmail.com";//change receiver address
$subject = "This is subject";
$message = "This is simple text message.";
$header = "From:nimu725@gmail.com \r\n";
$result = mail ($to,$subject,$message,$header);
if( $result == true ){
    echo "Message sent successfully...";
}else{
    echo "Sorry, unable to send mail...";
}
?>
```

Q3) Explain about File upload process in php?

- PHP allows you to upload single and multiple files through few lines of code only.
- PHP file upload features allows you to upload binary and text files both. Moreover, you can have the full control over the file to be uploaded through PHP authentication and file operation functions.

PHP File Upload Example:-

File: uploadform.html

```
<form action="uploader.php" method="post" enctype="multipart/form-data">
```

Select File:

```
<input type="file" name="fileToUpload"/>
```

```
<input type="submit" value="Upload Image" name="submit"/>
```

```
</form>
```

File: uploader.php

```
<?php
$target_path = "e/";
$target_path = $target_path.basename( $_FILES['fileToUpload']['name']);

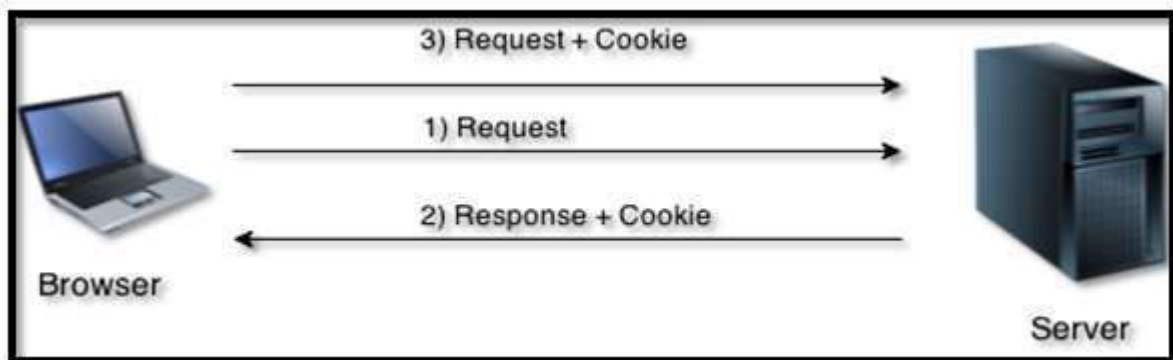
if(move_uploaded_file($_FILES['fileToUpload']['tmp_name'], $target_path)) {
    echo "File uploaded successfully!";
} else{
    echo "Sorry, file not uploaded, please try again!";
}
?>
```

Q4) Explain about Working on cookies in PHP?Q5)

PHP Cookie:-

PHP cookie is a small piece of information which is stored at client browser. It is used to recognize the user.

Cookie is created at server side and saved to client browser. Each time when client sends request to the server, cookie is embedded with request. Such way, cookie can be received at the server side.



In short, cookie can be created, sent and received at server end.

Syntax:

```
setcookie("CookieName", "CookieValue");/* defining name and value only*/
```

PHP \$_COOKIE:-

PHP \$_COOKIE superglobal variable is used to get cookie.

Example

`$value=$_COOKIE["CookieName");//returns cookie value`

PHP Cookie Example

File: cookie1.php

```
<?php
setcookie("user", "Murali");
?>
<html>
<body>
<?php
if(!isset($_COOKIE["user"])) {
    echo "Sorry, cookie is not found!";
} else {
    echo "<br/>Cookie Value: " . $_COOKIE["user"];
}
?>
</body>
</html>
```

Output:

Sorry, cookie is not found!

Firstly cookie is not set. But, if you refresh the page, you will see cookie is set now.

Output:

Cookie Value: Murali

PHP Delete Cookie:-

If you set the expiration date in past, cookie will be deleted.

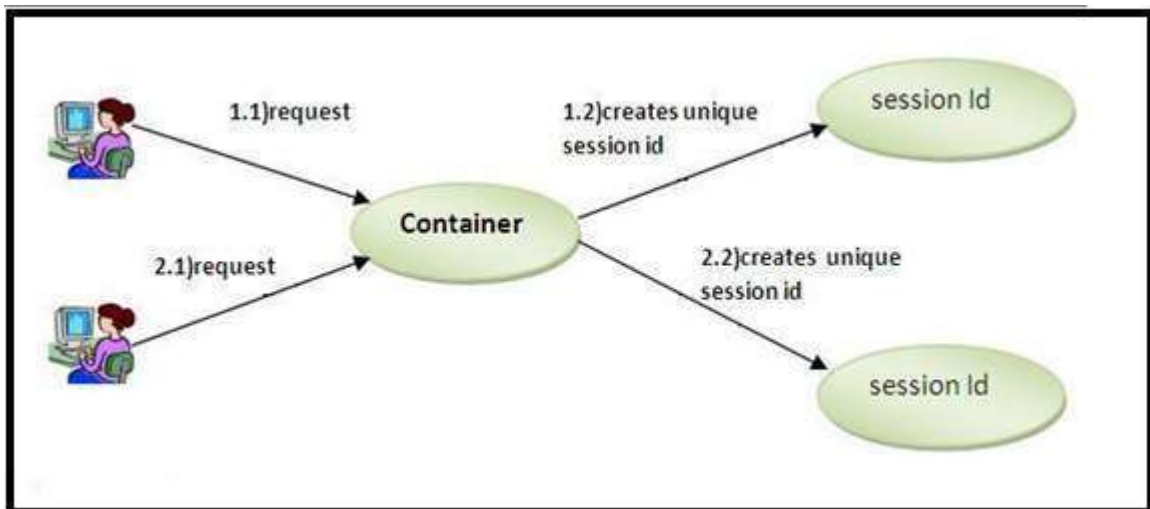
File: cookie1.php

```
<?php
setcookie ("CookieName", "", time() - 3600);// set the expiration date to one hour ago
?>
```

Q5) Explain about User sessions in PHP:-

PHP Session:-

- PHP session is used to store and pass information from one page to another temporarily (until user close the website).
- PHP session technique is widely used in shopping websites where we need to store and pass cart information e.g. username, product code, product name, product price etc from one page to another.
- PHP session creates unique user id for each browser to recognize the user and avoid conflict between multiple browsers.



PHP session start() function:-

PHP session_start() function is used to start the session. It starts a new or resumes existing session. It returns existing session if session is created already. If session is not available, it creates and returns new session.

Syntax:

```
bool session_start ( void )
```

Example:

```
session_start();
```

PHP \$_SESSION:-

PHP \$_SESSION is an associative array that contains all session variables. It is used to set and get session variable values.

Example: Store information

```
$_SESSION["user"] = "Sachin";
```

Example: Get information

```
echo $_SESSION["user"];
```

PHP Session Example

File: session1.php

```
<?php
session_start();
?>
<html>
<body>
<?php
$_SESSION["user"] = "Sachin";
echo "Session information are set successfully.<br/>";
?>
<a href="session2.php">Visit next page</a>
</body>
</html>
```

File: session2.php

```
<?php
session_start();
?>
<html>
<body>
<?php
echo "User is: ".$_SESSION["user"];
?>
</body>
</html>
```

Q6) Combining PHP in HTML code in single page:-

PHP in HTML:-

When building a complex page, at some point you will be faced with the need to combine PHP and HTML to achieve your needed results. At first point, this can seem complicated, since PHP and HTML are two separate languages, but this is not the case. PHP is designed to interact with HTML and PHP [scripts](#) can be included in an HTML page without a problem.

In an HTML page, PHP code is enclosed within special PHP tags. When a visitor opens the page, the server processes the PHP code and then sends the output (not the PHP code itself) to the visitor's browser. Actually it is quite simple to integrate HTML and PHP. A PHP

script can be treated as an HTML page, with bits of PHP inserted here and there. Anything in a PHP script that is not contained within <?php ?> tags is ignored by the PHP compiler and passed directly to the web browser.

If you look at the example below you can see what a full PHP script might look like:

EX:-

```
<html>
<head></head>
<body class="page_bg">
Hello, today is
<?php
echo date('M, d, Y, H, i, sa');
?>
</body>
</html>
```

Example:-

```
<html>
<head></head>
<body>
<ul>
<?php
for($i=1;$i<=5;$i++)
{
?>
<li>Menu Item
    <?php echo $i; ?>
</li>
<?php
}
?>
</ul>
</body>
</html>
```

End of unit-III

Unit-4:: Working with Files and Directories, Images

Q1) How to include files with include():-

Including a PHP File into Another PHP File:

PHP allows you to include file so that a page content can be reused many times. There are two ways to include file in PHP.

- **include**
- **require**

Advantage

Code Reusability: By the help of include and require construct, we can reuse HTML code or PHP script in many PHP scripts.

PHP include example

PHP include is used to include file on the basis of given path. Let's see a simple PHP include example.

File: footer.php

```
<?php  
Echo "copy right by N Murali Krishna Mtech ";  
?>
```

File: include1.html

```
<html>  
<body>  
<h1>Welcome to my home page!</h1>  
<p>Some text.</p>  
<p>Some more text.</p>  
<?php include 'footer.php';?>  
</body>  
</html>
```

require():-

The **include()** and **require()** statement allow you to include the code contained in a PHP file within another PHP file. Including a file produces the same result as copying the script from the file specified and pasted into the location where it is called.

You can save a lot of time and work through including files — Just store a block of code in a separate file and include it wherever you want by include() and require() statement instead of typing the entire block of code multiple times.

```
<?php require "footer.php"; ?>
<html>
<head>
    <title>mypage</title>
</head>
<body>
    <h1>Welcome to Our Website!</h1>
    <p>Here you will find lots of useful information.</p>
<?php include "footer.php"; ?>
</body>
</html>
```

Difference Between Include and Require Statements

if we can include files using the include() statement then why we need require(). Typically the require() statement operates like include().

The only difference is — the include() statement will only generate a PHP warning but allow script execution to continue if the file to be included can't be found, whereas the require() statement will generate a fatal error and stops the script execution.

Q2) Working with Files in PHP? Explain about file modes?

PHP is a server side programming language, it allows you to work with files and directories stored on the web server. We can create, access, and manipulate files on your web server using the PHP file system functions.

- **fopen()**
- **fread()**
- **fwrite()**
- **rename()**
- **fclose()**

Opening a File with PHP fopen() Function

To work with a file you first need to open the file. The PHP fopen() function is used to open a file.

The basic syntax:-

fopen(filename, mode)

The first parameter passed to fopen() specifies the name of the file you want to open, and the second parameter specifies in which mode the file should be opened. For example:

```
<?php
$handle = fopen("data.txt", "r");
if($handle){
    echo "File opened successfully.";

    // Closing the file handle
    fclose($handle);
}
?>
```

The file may be opened in one of the following modes:

Modes	What it does
r	Open the file for reading only.
r+	Open the file for reading and writing.
w	Open the file for writing only and clears the contents of file. If the file does not exist, PHP will attempt to create it.
w+	Open the file for reading and writing and clears the contents of file. If the file does not exist, PHP will attempt to create it.
a	Append. Opens the file for writing only. Preserves file content by writing to the end of the file. If the file does not exist, PHP will attempt to create it.
a+	Read/Append. Opens the file for reading and writing. Preserves file content by writing to the end of the file. If the file does not exist, PHP will attempt to create it.

Reading from Files with PHP fread() Function

Now that you have understood how to open and close files. In the following section you will learn how to read data from a file. PHP has several functions for reading data from a file. You can read from just one character to the entire file with a single operation. Reading Fixed Number of Characters

The fread() function can be used to read a specified number of characters from a file.

The basic syntax:-

fread(file handle, length in bytes)

```
<?php
$file = "data.txt";
// Check the existence of file
if(file_exists($file)){
    // Open the file for reading
    $handle = fopen($file, "r") or die("ERROR: Cannot open the file.");

    // Read fixed number of bytes from the file
    $content = fread($handle, "20");

    // Closing the file handle
    fclose($handle);

    // Display the file content
    echo $content;
} else{
    echo "ERROR: File does not exist.";
}
?>
```

Writing the Files Using PHP fwrite() Function

Similarly, you can write data to a file or append to an existing file using the PHP fwrite() function.

The basic syntax:-

fwrite(file handle, string)

The fwrite() function takes two parameter — A file handle and the string of data that is to be written, as demonstrated in the following example:

```
<?php
$file = "note.txt";

// String of data to be written
$data = "hi i am a lecturer in shdc";

// Open the file for writing
$handle = fopen($file, "w");

// Write data to the file
fwrite($handle, $data);

// Closing the file handle
fclose($handle);

echo "Data written to the file successfully.";
?>
```

Renaming Files with PHP rename() Function:

you can rename a file or directory using the PHP's rename() function, like this:

```
<?php
$file = "note.txt";

// Check the existence of file
if(file_exists($file)){
    // Attempt to rename the file
    if(rename($file, "newfile.txt")){
        echo "File renamed successfully.";
    } else{
        echo "ERROR: File cannot be renamed.";
    }
} else{
    echo "ERROR: File does not exist.";
}
?>
```

Closing a File with PHP fclose() Function

Once you've finished working with a file, it needs to be closed. The fclose() function is used to close the file, as shown in the following example:

```
<?php
$file = "data.txt";

// Check the existence of file
if(file_exists($file)){
    // Open the file for reading
    $handle = fopen($file, "r");

    /* Some code to be executed */

    // Closing the file handle
    fclose($handle);
} else{
    echo "ERROR: File does not exist.";
}
?>
```

PHP Delete File - unlink():-

The PHP unlink() function is used to delete file.

Syntax

unlink (filename) ;

Example

```
<?php
unlink('data.txt');

echo "File deleted successfully";
?>
```

PHP Append to File:-

You can append data into file by using a or a+ mode in fopen() function. Let's see a simple example that appends data into data.txt file.

Let's see the data of file first.

data.txt

welcome to php file write

PHP Append to File - fwrite()

The PHP fwrite() function is used to write and append data into file.

Example

```
<?php
$fp = fopen('data.txt', 'a');//opens file in append mode
fwrite($fp, ' this is additional text ');
fwrite($fp, 'appending data');
fclose($fp);
echo "File appended successfully";
?>
```

Q3) Create a new directory in php (or) explain how to work with directories:-

Working with Directories in PHP:-

PHP also allows you to work with directories on the file system, for example, you can open a directory and read its contents, create or delete a directory, list all files in the directory, and so on.

Creating a New Directory

You can create a new and empty directory by calling the PHP mkdir() function with the path and name of the directory to be created, as shown in the example below:

```
<?php
// The directory path
$dir = "testdir";

// Check the existence of directory
if(!file_exists($dir)){
    // Attempt to create directory
    if(mkdir($dir)){
        echo "Directory created successfully.";
    } else{
        echo "ERROR: Directory could not be created.";
    }
} else{
    echo "ERROR: Directory already exists.";
}
?>
```

Q4) Copying Files from One Location to Another:-

We can copy a file from one location to another by calling PHP copy() function with the file's source and destination paths as arguments. If the destination file already exists it'll be overwritten. Here's an example which creates a copy of "example.txt" file inside backup folder.

```
<?php
// Source file path
$file = "example.txt";

// Destination file path
$newfile = "backup/example.txt";

// Check the existence of file
if(file_exists($file)){
    // Attempt to copy file
    if(copy($file, $newfile)){
        echo "File copied successfully.";
    } else{
        echo "ERROR: File could not be copied.";
    }
} else{
    echo "ERROR: File does not exist.";
}
?>
```

To make this example work, the target directory which is backup and the source file i.e. "example.txt" has to exist already; otherwise PHP will generate an error.

Q5) Explain about Running commands in PHP:-

System():-

system — Execute an external program and display the output.

Description

string system (string \$command [, int &\$amp;return_var])

Parameters :-

command:-

The command that will be executed.

return var:-

If the return_var argument is present, then the return status of the executed command will be written to this variable.

Return Values :-

Returns the last line of the command output on success, and FALSE on failure.

Examples

```
<?php
echo '<pre>';

// Outputs all the result of shellcommand "ls", and returns
// the last output line into $last_line. Stores the return value
// of the shell command in $retval.
$last_line = system('ls', $retval);

// Printing additional info
echo '
</pre>

<hr />Last line of the output: ' . $last_line . '
<hr />Return value: ' . $retval;
?>
```

Exec():-

exec — Execute an external program.

Description:-

string exec (string \$command [, array &\$output [, int &\$return_var]])

exec() executes the given command.

```
<?php
// outputs the username that owns the running php/httpd process
// (on a system with the "whoami" executable in the path)
echo exec('whoami');
?>
```

Passthru():-

passthru — Execute an external program and display raw output.

Description

void passthru (string \$command [, int &\$return_var])

example:-

```
<?php
$filename = "backup-" . date("d-m-Y");
$cmd = "mysqldump -u root dudh_society >c:/Backup/$filename.sql";
passthru( $cmd );
if(passthru($cmd) == true)
{
    echo "Backup Succesfully";
}
else
{
    echo "Backup failed";
}
?>
```

Q6) Working with images in php:-

Before we begin drawing on our image, there are two functions that we should consider, for added variety.

1. Line color can be modified using the `imagecolorallocate()` function, which we learned about before. It should be stored in a variable to be used later.
2. Line thickness can be modified using the `imagegetthickness()` function, which requires two parameters: `imagegetthickness(image, thickness)`

The `imageline()` function itself requires 6 parameters.

The syntax is:

`imageline(image, x1, y1, x2, y2, color)`

image = Refers to the Image Resource That the Line Will Be Applied to

x1 = x-coordinate For First Point

y1 = y-coordinate For First Point

x2 = x-coordinate For Second Point

y2 = y-coordinate For Second Point

color = Refers to the Line Color Identifier Created With `imagecolorallocate()`

create a 200x200 square (and a bit more)

```
<?php
create_image();
echo "<img src=image.png?".date("U").">";

function create_image()
{
    $im = @imagecreate(200, 200) or die("Cannot Initialize new GD image stream");
    $background_color = imagecolorallocate($im, 255, 255, 0); // yellow
    imagepng($im,"image.png");
    imagedestroy($im);
}
?>
```

Draw line:-

```
<?php
create_image();
print "<img src=image.png?".date("U").">";

function create_image(){
    $im = @imagecreate(200, 200) or die("Cannot Initialize new GD image stream");
    $background_color = imagecolorallocate($im, 255, 255, 0); // yellow

    $red = imagecolorallocate($im, 255, 0, 0);           // red
    $blue = imagecolorallocate($im, 0, 0, 255);          // blue
    imageline ($im, 5, 5, 195, 5, $red);
    imageline ($im, 5, 5, 195, 195, $blue);

    imagepng($im,"image.png");
    imagedestroy($im);
}
?>
```

[imageline](#)(\$im, X₁, Y₁, X₂, Y₂, \$color):

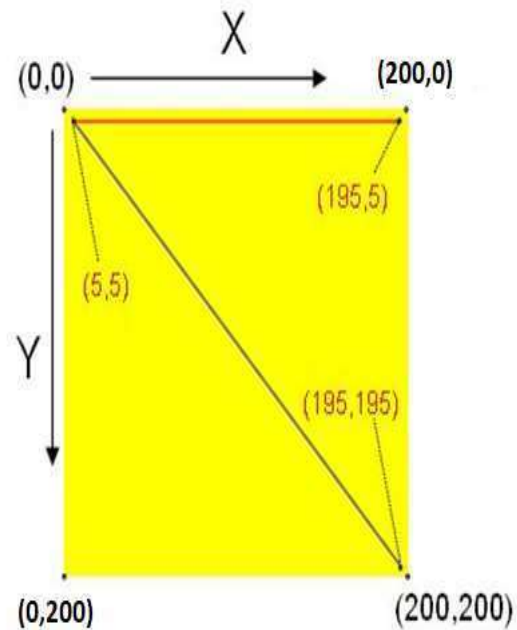
where X₁, Y₁, X₂ and Y₂ are the positions within the image.

For example, the red line starts at

X₁=5 and Y₁=5

and ends at

X₂=195 and Y₂=5



Draw rectangle:-

```
<?php
create_image();
print "<img src=image.png?".date("U").">";

function create_image(){
    $im = @imagecreate(200, 200) or die("Cannot Initialize new GD image stream");
    $background_color = imagecolorallocate($im, 255, 255, 0); // yellow

    $red = imagecolorallocate($im, 255, 0, 0); // red
    $blue = imagecolorallocate($im, 0, 0, 255); // blue
    imagerectangle ($im, 5, 10, 195, 50, $red);
    imagefilledrectangle ($im, 5, 100, 195, 140, $blue);

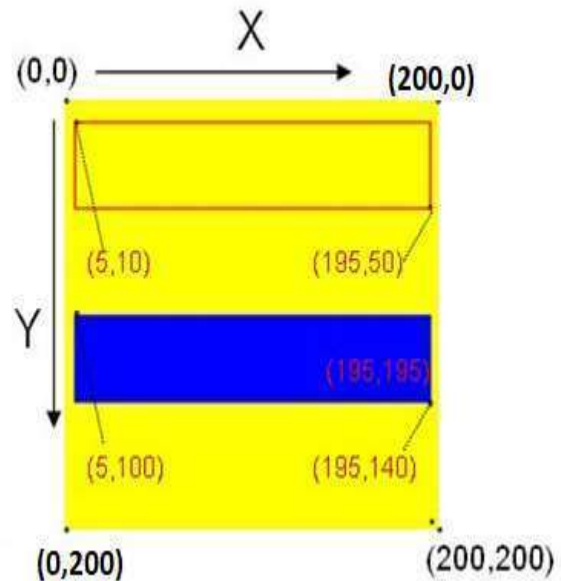
    imagepng($im,"image.png");
    imagedestroy($im);
}
?>
```

```

imagerectangle($im, $x1, $y1, $x2, $y2, $color);
imagefilledrectangle($im, $x1, $y1, $x2, $y2, $color);

```

For example, the red rectangle starts at
 $X_1=5$ and $Y_1=10$
 and ends at
 $X_2=195$ and $Y_2=50$



Draw ellipse:-

```

<?php
create_image();
print "<img src=image.png?".date("U").">";

function create_image(){
    $im = @imagecreate(200, 200) or die("Cannot Initialize new GD
image stream");
    $background_color = imagecolorallocate($im, 255, 255, 0); //
yellow

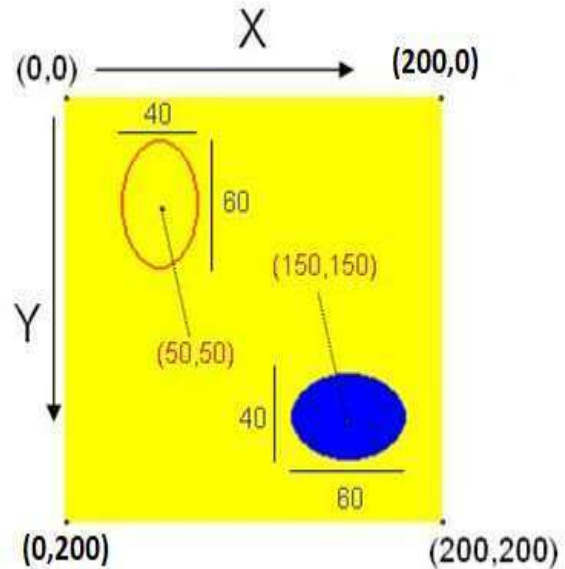
    $red = imagecolorallocate($im, 255, 0, 0);           // red
    $blue = imagecolorallocate($im, 0, 0, 255);          // blue
    imageellipse($im, 50, 50, 40, 60, $red);
    imagefilledellipse($im, 150, 150, 60, 40, $blue);

    imagepng($im,"image.png");
    imagedestroy($im);
}
?>

```

```
imageellipse($im, X, Y, width, height, $color);
imagefilledellipse($im, X, Y, width, height, $color);
```

For example, the center of the red ellipse is located at
 X=50
 Y=50
 and the dimensions are
 width = 40
 height= 60



Add text to the image:-

```
<?php
create_image();
print "<img src=image.png?".date("U").">";

function create_image(){
    $im = @imagecreate(200, 200) or die("Cannot Initialize new GD image stream");
    $background_color = imagecolorallocate($im, 255, 255, 0); // yellow
    $red = imagecolorallocate($im, 255, 0, 0); // red

    imagestring($im, 1, file:///C:/apache2triad/htdocs/0image/tutorial1.html 5, 10,
    "Hello !", $red);
    imagestring($im, 2, 5, 50, "Hello !", $red);
    imagestring($im, 3, 5, 90, "Hello !", $red);
    imagestring($im, 4, 5, 130, "Hello !", $red);
    imagestring($im, 5, 5, 170, "Hello !", $red);

    imagestringup($im, 5, 140, 150, "Hello !", $red);

    imagepng($im, "image.png");
    imagedestroy($im);
}
?>
```

output:-



Hello !

Hello !

Hello !

Hello !

Hello !

Hello !

End of unit-IV

Unit-V :: Interacting with MySQL using PHP

Q1) Explain about MySQL Versus MySQLi Functions:-

MySQL MySQLi Differences:-

There are too many differences between these PHP database extensions. These differences are based on some factors like performance, library functions, features, benefits, and others.

MySQL	MySQLi
MySQL extension added in PHP version 2.0. and deprecated as of PHP 5.5.0.	MySQLi extension added in PHP 5.5 and will work on MySQL 4.1.3 or above.
Does not support prepared statements.	MySQLi supports prepared statements.
MySQL provides the procedural interface.	MySQLi provides both procedural and object-oriented interface.
MySQL extension does not support stored procedure.	MySQLi supports store procedure.
MySQL extension lags in security and other special features, comparatively.	MySQLi extension is with enhanced security and improved debugging.
Transactions are handled by SQL queries only.	MySQLi supports transactions through API.
Extension directory: ext/mysql.	Extension directory: ext/mysqli.

Different functions of MySQL functions:-

MySQL supports several(built-in) functions like mathematical functions ,text functions, string functions, date and time functions, advanced functions and encrypted

Different functions of MySQL:-

MySQL Arithmetic Functions

MySQL provides a wide range of functions designed to serve a range of arithmetical purposes.

- ABS() — Returns the absolute value
- COS() — Returns the cosine
- COT() — Returns the cotangent
- DIV() — Integer division
- EXP() — Returns to the power of the argument
- MOD() — Returns the remainder
- POW(), POWER() — Returns the argument raised to the specified power
- ROUND() — Rounds the argument
- SIN() — Returns the sine of the argument
- SQRT() — Returns the square root of the argument
- TAN() — Returns the tangent of the argument
- TRUNCATE()—Truncates to specified number of decimal places specified in the argument

MySQL aggregate functions

- AVG – calculate the average value of a set of values or an expression.
- COUNT – count the number rows in a table.
- INSTR – return the position of the first occurrence of a substring in a string.
- SUM – calculate the sum of a set of values or an expression.
- MIN – find the minimum value in a set of values
- MAX – find the maximum value in a set of values

MySQL string functions

- CONCAT – combine two or more strings into one string.
- LENGTH & CHAR_LENGTH – get the length of strings in bytes and in characters.
- LEFT – get the left part of a string with a specified length.
- REPLACE – search and replaces a substring in a string.
- SUBSTRING – extract a substring starting from a position with a specific length.

·TRIM – remove unwanted characters from a string.

MySQL date and time functions

- CURDATE – return the current date.
- DAY – get the day of the month of a specified date.
- SYSDATE – return the current date.
- WEEK – return a week number of a date.
- WEEKDAY – return a weekday index for a date.
- YEAR – return the year part of a date value.

MySQLi-Functions:-

PHP MySQLi = PHP MySQL Improved!

The MySQLi functions allows you to access MySQL database servers.

Note: The MySQLi extension is designed to work with MySQL version 4.1.13 or newer.

Q2) How to connect mysql with PHP:-

PHP MySQL Connect:-

Since PHP 5.5, mysql_connect() extension is deprecated. Now it is recommended to use one of the 2 alternatives.

- 1)mysqli_connect()
- 2)PDO::__construct()

PHP mysqli_connect()

PHP mysqli_connect() function is used to connect with MySQL database. It returns resource if connection is established or null.

Syntax

resource mysqli_connect (server, username, password)

PHP mysqli_close()

PHP mysqli_close() function is used to disconnect with MySQL database. It returns true if connection is closed or false.

Syntax

bool mysqli_close(resource \$resource_link)

PHP MySQL Connect Example:-

Example

```
<?php
$host = 'localhost:3306';
$user = '';
$pass = '';
$conn = mysqli_connect($host, $user, $pass);
if(! $conn )
{

    die('Could not connect: ' . mysqli_error());

}
echo 'Connected successfully';
mysqli_close($conn);
?>
```

Connecting to MySQL Database Server:-

In PHP you can easily do this using the `mysqli_connect()` function. All communication between PHP and the MySQL database server takes place through this connection. Here're the basic syntaxes for connecting to MySQL using MySQLi and PDO extensions:

Syntax: MySQLi, Procedural way

```
$link = mysqli_connect("hostname", "username", "password", "database");
```

Syntax: MySQLi, Object Oriented way

```
$mysqli = new mysqli("hostname", "username", "password", "database");
```

Syntax: PHP Data Objects (PDO) way

```
$pdo = new PDO("mysql:host=hostname;dbname=database", "username", "password");
```

The hostname parameter in the above syntax specify the host name (e.g. localhost), or IP address of the MySQL server, whereas the username and password parameters specifies the credentials to access MySQL server, and the database parameter, if provided will specify the default MySQL database to be used when performing queries.

```
<?php
/* Attempt MySQL server connection. Assuming you are running MySQL
server with default setting (user 'root' with no password) */
$link = mysqli_connect("localhost", "root", "");

// Check connection
if($link === false){
    die("ERROR: Could not connect. " . mysqli_connect_error());
}

// Print host information
echo "Connect Successfully. Host info: " . mysqli_get_host_info($link);
?>
```

Q3) Create a MySQL Database Using PHP:-

Creating MySQL Database:-

Before saving or accessing the data, we need to create a database first. The [CREATE DATABASE](#) statement is used to create a new database in MySQL.

Let's make a SQL query using the CREATE DATABASE statement, after that we will execute this SQL query through passing it to the PHP `mysqli_query()` function to finally create our database. The following example creates a database named demo.

Example:-

```
<?php
/* Attempt MySQL server connection. Assuming you are running MySQL
server with default setting (user 'root' with no password) */
$link = mysqli_connect("localhost", "root", "");

// Check connection
if($link === false){
    die("ERROR: Could not connect. " . mysqli_connect_error());
}

// Attempt create database query execution
$sql = "CREATE DATABASE demo";
if(mysqli_query($link, $sql)){
    echo "Database created successfully";
} else{
    echo "ERROR: Could not able to execute $sql. " . mysqli_error($link);
}
```

```
}
```

```
// Close connection  
mysqli_close($link);  
?>
```

Q4) Insert a record into the table using PHP:-

The [INSERT INTO](#) statement is used to insert new rows in a database table. Let's make a SQL query using the INSERT INTO statement with appropriate values, after that we will execute this insert query through passing it to the PHP `mysqli_query()` function to insert data in table. Here's an example, which insert a new row to the persons table by specifying values for the first_name, last_name and email fields.

Example:

```
<?php  
/* Attempt MySQL server connection. Assuming you are running MySQL  
server with default setting (user 'root' with no password) */  
$link = mysqli_connect("localhost", "root", "", "demo");  
  
// Check connection  
if($link === false){  
    die("ERROR: Could not connect. " . mysqli_connect_error());  
}  
  
// Attempt insert query execution  
$sql = "INSERT INTO persons (first_name, last_name, email) VALUES ('Peter', 'Parker',  
'peterparker@mail.com')";  
if(mysqli_query($link, $sql)){  
    echo "Records inserted successfully.";  
} else{  
    echo "ERROR: Could not able to execute $sql. " . mysqli_error($link);  
}  
  
// Close connection  
mysqli_close($link);  
?>
```

Q5) Create a Mysql database table(person address table) using PHP:-

We can create tables inside the database that will actually hold the data. A table organizes the information into rows and columns.

The SQL [CREATE TABLE](#) statement is used to create a table in database.

Let's make a SQL query using the CREATE TABLE statement, after that we will execute this SQL query through passing it to the PHP `mysqli_query()` function to finally create our table.

Example:-

```
<?php
/* Attempt MySQL server connection. Assuming you are running MySQL
server with default setting (user 'root' with no password) */
$link = mysqli_connect("localhost", "root", "", "demo");

// Check connection
if($link === false){
    die("ERROR: Could not connect. " . mysqli_connect_error());
}

// Attempt create table query execution
$sql = "CREATE TABLE persons(
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    first_name VARCHAR(30) NOT NULL,
    last_name VARCHAR(30) NOT NULL,
    email VARCHAR(70) NOT NULL UNIQUE
    Door_no int(10),
    Street_name varchar(30),
    Village varchar(30),
    Mandal varchar(30),
    State varchar(30),
    pin int(6)
)";
if(mysqli_query($link, $sql))
{
    echo "Table created successfully.";
}
else
{
    echo "ERROR: Could not able to execute $sql. " . mysqli_error($link);
}
```

```
// Close connection
mysqli_close($link);
?>
```

Q6) Explain how to view records from the table using PHP.

View/Selecting Data From Database Tables

The SQL [SELECT](#) statement is used to select the records from database tables. Its basic syntax is as follows:

SELECT column1_name, column2_name, columnN_name FROM table_name;

Let's make a SQL query using the SELECT statement, after that we will execute this SQL query through passing it to the PHP `mysqli_query()` function to retrieve the table data.

Consider our persons database table has the following records:

```
+---+-----+-----+-----+
| id | first_name | last_name | email          |
+---+-----+-----+-----+
| 1 | Peter      | Parker    | peterparker@mail.com |
| 2 | John       | Rambo     | johnrambo@mail.com   |
| 3 | Clark      | Kent      | clarkkent@mail.com   |
| 4 | John       | Carter    | johncarter@mail.com  |
| 5 | Harry      | Potter    | harrypotter@mail.com  |
+---+-----+-----+-----+
```

The PHP code in the following example selects all the data stored in the persons table (using the asterisk character (*) in place of column name selects all the data in the table).

Example:

```
<?php
/* Attempt MySQL server connection. Assuming you are running MySQL
server with default setting (user 'root' with no password) */
$link = mysqli_connect("localhost", "root", "", "demo");

// Check connection
if($link === false){
    die("ERROR: Could not connect. " . mysqli_connect_error());
}
```

```

// Attempt select query execution
$sql = "SELECT * FROM persons";
if($result = mysqli_query($link, $sql)){
    if(mysqli_num_rows($result) > 0){
        echo "<table>";
        echo "<tr>";
        echo "<th>id</th>";
        echo "<th>first_name</th>";
        echo "<th>last_name</th>";
        echo "<th>email</th>";
        echo "</tr>";
        while($row = mysqli_fetch_array($result)){
            echo "<tr>";
            echo "<td>" . $row['id'] . "</td>";
            echo "<td>" . $row['first_name'] . "</td>";
            echo "<td>" . $row['last_name'] . "</td>";
            echo "<td>" . $row['email'] . "</td>";
            echo "</tr>";
        }
        echo "</table>";
        // Free result set
        mysqli_free_result($result);
    } else{
        echo "No records matching your query were found.";
    }
} else{
    echo "ERROR: Could not able to execute $sql. " . mysqli_error($link);
}

// Close connection
mysqli_close($link);
?>

```

Q7)How to delete table record using PHP.(Delete Mechanism)

Deleting Database Table Data

We can delete records from a table using the SQL [DELETE](#) statement. It is typically used in conjugation with the WHERE clause to delete only those records that matches specific criteria or condition.

The basic syntax of the DELETE statement can be given with:

DELETE FROM table_name WHERE column_name=some_value

Let's make a SQL query using the DELETE statement and WHERE clause, after that we will execute this query through passing it to the PHP mysqli_query() function to delete the tables records.

Consider the following persons table inside the demo database:

id	first_name	last_name	email
1	Peter	Parker	peterparker@mail.com
2	John	Rambo	johnrambo@mail.com
3	Clark	Kent	clarkkent@mail.com
4	John	Carter	johncarter@mail.com
5	Harry	Potter	harrypotter@mail.com

The PHP code in the following example will delete the records of those persons from the persons table whose first_name is equal to John.

Example Program:-

```
<?php
/* Attempt MySQL server connection. Assuming you are running MySQL
server with default setting (user 'root' with no password) */
$link = mysqli_connect("localhost", "root", "", "demo");

// Check connection
if($link === false){
    die("ERROR: Could not connect. " . mysqli_connect_error());
}

// Attempt delete query execution
$sql = "DELETE FROM persons WHERE first_name='John'";
if(mysqli_query($link, $sql)){
    echo "Records were deleted successfully.";
} else{
    echo "ERROR: Could not able to execute $sql. " . mysqli_error($link);
}

// Close connection
mysqli_close($link);
?>
```

Q8) Explain about how to adding sub-entities to a record?

Parent-child relationship in MySQL has to deal with establishing some hierarchical dependencies between records in a database table. In business terms, it will mean establishing a dependency relationship between two similar entities. It is used to specify where one table's value refers to another value in some other table (usually a primary key of another table).

For example, imagine a book store. This could be represented by a table called books and has fields for id, title, publisher and author. But instead of placing a name in the author column, you could instead put the ID value of a author in a separate table called authors that has information on authors such as id, name, and email.

Therefore, if you need to update an author's email, you only need to do so in single (authors (parent)) table, instead in multiple rows of books table.

This concept in dataBase is commonly known as 'Foreign key', where parent-child relation is 1:N type association between the tables. Each child points to its parent. And each child has exactly one parent. A parent entity contains no information as to who its children are.

*******THE END*******