UNIVERSITY OF PUNE

LAB COURSE II

WEB DEVELOPMENT AND PHP PROGRAMMING - I (COURSE CODE:CS-348)

T.Y.B.SC.(COMPUTER SCIENCE)

SEMESTER I

ADVISORS:

PROF. A. G. GANGARDE (CHAIRMAN, BOS-COMP. SC.)

CHAIRMAN:

PROF. MRS. CHITRA NAGARKAR

CO-ORDINATOR:

PROF. MRS. SHAIKH A.M.

MEMBERS:

SHINDE SAHEBRAO
NIKAM PRADEEP
JOSHI GAJANAN
MAHAJAN NILESH
RAUT SANGEETA
MUSALE VAIBHAV
SATPUTE JAYA
TANK BHUPESH
MANKAR ABHIJEET
JOSHI VARSHA
MAHAJAN MANISHA
RAYATE RUPALI

BOARD OF STUDY (COMPUTER SCIENCE) MEMBERS:

1. MR. M. N. SHELAR
2. MR. S. N. SHINDE
3. MR. U. S. SURVE
4. MR. V. R. WANI
5. MR. PRASHANT MULE
6. DR.VILAS KHARAT
7. MRS. CHITRA NAGARKAR
8. MR. S. S. DESHMUKH

ABOUT THE WORK BOOK

OBJECTIVES OF THIS BOOK

THIS WORKBOOK IS INTENDED TO BE USED BY T.Y.B.SC(COMPUTER SCIENCE) STUDENTS FOR THE THREE COMPUTER SCIENCE LABORATORY COURSES.

THE OBJECTIVES OF THIS BOOK ARE

1. THE SCOPE OF THE COURSE.
2. BRINGING UNIFORMITY IN THE WAY COURSE IS CONDUCTED ACROSS DIFFERENT COLLEGES.
3. CONTINUOUS ASSESSMENT OF THE STUDENTS.
4. PROVIDING READY REFERENCES FOR STUDENTS WHILE WORKING IN THE LAB.

HOW TO USE THIS BOOK?

THIS BOOK IS MANDATORY FOR THE COMPLETION OF THE LABORATORY COURSE. IT IS A MEASURE OF THE PERFORMANCE OF THE STUDENT IN THE LABORATORY FOR THE ENTIRE DURATION OF THE COURSE.

INSTRUCTIONS TO THE STUDENTS

1) STUDENTS SHOULD CARRY THIS BOOK DURING PRACTICAL SESSIONS OF COMPUTER SCIENCE.

2) STUDENTS SHOULD MAINTAIN SEPARATE JOURNAL FOR THE SOURCE CODE AND OUTPUTS.

3) STUDENT SHOULD READ THE TOPICS MENTIONED IN **READING SECTION** OF THIS BOOK BEFORE COMING FOR PRACTICAL.

4) STUDENTS SHOULD SOLVE ONLY THOSE EXERCISES WHICH ARE SELECTED BY PRACTICAL IN-CHARGE AS A PART OF JOURNAL ACTIVITY. HOWEVER, STUDENTS ARE FREE TO SOLVE ADDITIONAL EXERCISES TO DO MORE PRACTICE FOR THEIR PRACTICAL EXAMINATION.

5) STUDENTS WILL BE ASSESSED FOR EACH EXERCISE ON A SCALE OF 5

1.NOTE DONE 0
2. INCOMPLETE 1
3.LATE COMPLETE 2
4.NEEDS IMPROVEMENT 3
5.COMPLETE 4
6.WELLDONE 5

INSTRUCTIONS TO THE PRACTICAL IN-CHARGE

1) EXPLAIN THE ASSIGNMENT AND RELATED CONCEPTS IN AROUND TEN MINUTES USING WHITE BOARD IF REQUIRED OR BY DEMONSTRATING THE SOFTWARE.

2) CHOOSE APPROPRIATE PROBLEMS TO BE SOLVED BY STUDENT.

3) AFTER A STUDENT COMPLETES A SPECIFIC SET, THE INSTRUCTOR HAS TO VERIFY THE OUTPUTS AND SIGN IN THE PROVIDED SPACE AFTER THE ACTIVITY.

4) ENSURE THAT THE STUDENTS USE GOOD PROGRAMMING PRACTICES.

5) YOU SHOULD EVALUATE EACH ASSIGNMENT CARRIED OUT BY A STUDENT ON A SCALE OF 5 AS SPECIFIED ABOVE TICKING APPROPRIATE BOX.

6) THE VALUE SHOULD ALSO BE ENTERED ON ASSIGNMENT COMPLETION PAGE OF RESPECTED LAB COURSE.

ASSIGNMENT NO. 1 : TO STUDY FUNCTIONS & STRINGS.

User-defined functions

A function may be defined using syntax such as the following:

```
function function_name([argument_list…])
{
    [statements]
    [return return_value;]
}
```

Any valid PHP code may appear inside a function, even other functions and class definitions. The variables you use inside a function are, by default, not visible outside that function. In PHP3 functions must be defined, before they are referenced. No such requirement exists in PHP4.

Example 1.

Code	**Output**

<?php
msg("Hello"); // calling a function
function msg($a) // defining a function
{
echo $a;
}
?>
	Hello

Default parameters

You can give default values to more than one argument, but once you start assigning default values, you have to give them to all arguments that follow as well.

Example 2.

Code	Output

<?php
function display($greeting, $message="Good Day")
{
  echo $greeting;
  echo "
";
 echo $message;
}
display("Hello");
?>
	Hello
	Good Day

Variable parameters

You can set up functions that can take a variable number of arguments. Variable number of arguments can be handled with these functions:

func_num_args : Returns the number of arguments passed

func_get_arg : Returns a single argument

func_get_args : Returns all arguments in an array

Example 3.

Code	Output

<?php
echo "Passing 3 arg. to xconcat
";
echo "Result is …";
xconcat("How","are","you");
function xconcat( )
{
  $ans = "";
  $arg = func_get_args( );
  for ($i=0; $i<func_num_args( ); $i++)
	Passing 3 arg. to xconcat
	Result is …How are you

{
$ans .= $arg[$i].” “;
}
echo $ans;
}
?>	

Missing parameters

When using default arguments, any defaults should be on the right side of any non-default arguments, otherwise, things will not work as expected.

Example 4.

Code	**Output**

<?php
function makecoffee ($type = "Nescafe")
{
return "Making a cup of $type
";
}
echo makecoffee ();
echo makecoffee ("espresso");
?>
	Making a cup of Nescafe.
	Making a cup of espresso.

<?php
function make ($type = "acidophilus", $flavour)
{
return "Making a bowl of $type $flavour
";
}
echo make ("raspberry"); // won’t work
?>
	Warning: Missing argument 2 in call to make()…..
	Making a bowl of raspberry

<?php
function make ($flavour, $type = "acidophilus")
{
return "Making a bowl of $type $flavour
";
}
echo make ("raspberry"); //it works
?>
	Making a bowl of acidophilus raspberry.

Variable functions

Assign a variable the name of a function, and then treat that variable as though it is the name of a function.

Example 5.

Code	**Output**

<?php
$varfun='fun1';
$varfun( );
$varfun='fun2';
$varfun( );
$varfun='fun3';
$varfun( );
function fun1( )
{
echo "
Function one";
}
function fun2( )
{
echo "
Function two";
}
function fun3( )
{
echo "
Function three";
}
?>
	Function one
	Function two
	Function three

Anonymous functions

The function that does not possess any name are called anonymous functions. Such functions are created using *create_function( )* built-in function. Anonymous functions are also called as lambda functions.

Example 6.

Code	**Output**
<?php $fname=create_function('$a,$b', '$c = $a + $b; return $c;'); echo $fname(10,20); ?>	

Strings

Strings in PHP

- Single quoted string (few escape characters supported, variable interpolation not possible)
- Double quoted string (many escape characters supported, variable interpolation possible)
- Heredoc

There are functions to print the string, namely print, printf, echo.

The print statement can print only single value, whereas echo and printf can print multiple values. Printf requires format specifiers. If echo statement is used like a function, then only one value can be printed.

Comparing Strings

Example 1.

Code	Output

<?php
$a='amit';
$b='anil';
if($a==$b) //using operator
echo "Both strings are equal
";
else
echo "Both strings are not equal
";
if(strcmp($a,$b)>0) //using function
{
echo "String2 sorts before String1";
}
elseif(strcmp($a,$b)==0)
{
echo "both are equal";
}
elseif(strcmp($a,$b)<0) // negative value
{
echo "String1 sorts before String2";
}?>
	Both strings are not equal
	String1 sorts before String2

Other string comparison functions

strcasecmp( ) : case in-sensitive string comparison

strnatcmp( ) : string comparison using a “natural order” algorithm

strnatcasecmp( ) : case in-sensitive version of strnatcmp( )

String manipulation & searching string

Example 2.

Code	Output
<?php $small="India"; $big="India is my country"; $str=substr($big,6,5); echo " $str"; $cnt = substr_count($big,"i"); echo " There are".$cnt." i's in $big"; $pos=strpos($big,"is"); echo " is found at $pos position"; $replace=substr_replace($big,"Bharat",0,5); echo " before replacement->$big"; echo " after replacement ->$replace"; ?>	is my There are 2 i's in India is my country is found at 6 position before replacement->India is my country after replacement ->Bharat is my country

Regular Expressions

Two types of regular expressions

- POSIX – style
- PERL – compatible

Purpose of using regular expressions

- Matching
- Substituting
- Splitting

Example 3.

Code	Output
<?php $big=<<< paragraph India is my country. I am proud of it. I live in Maharashtra. paragraph; echo " "; $found=preg_match('/am/i',$big); if($found) echo " am found in \$big"; $replace=preg_replace('/India/','Bharat',$big); echo " $replace"; $split=preg_split('/ /',$big); foreach($split as $elem) { echo " $elem";} ?>	am found in $big Bharat is my country. I am proud of it. I live in Maharashtra. India is my country. I am proud of it. I live in Maharashtra

Set A

1. Write a PHP script for the following: Design a form to accept a string. Write a function to count the total number of vowels (a,e,i,o,u) from the string. Show the occurrences of each vowel from the string. Check whether the given string is a palindrome or not, without using built-in function. (Use radio buttons and the concept of function. Use ‘include’ construct or require stmt.)
2. Write a PHP script for the following: Design a form to accept two strings from the user. Find the first occurrence and the last occurrence of the small string in the large string. Also count the total number of occurrences of small string in the large string. Provide a text box to accept a string, which will replace the small string in the large string. (Use built-in functions)

Set B

1. Write a PHP script for the following: Design a form to accept two numbers from the user. Give options to choose the arithmetic operation (use radio buttons). Display the result on the next form. (Use the concept of function and default parameters. Use 'include' construct or require stmt)
2. Write a PHP script for the following: Design a form to accept two strings from the user. Find whether the small string appears at the start of the large string. Provide a text box to accept the string that will replace all occurrences of small string present in the large string. Also split the large string into separate words. (Use regular expressions)

Set C

1. Write a PHP script for the following: Design a form to accept the details of 5 different items, such as item code, item name, units sold, rate. Display the bill in the tabular format. Use only 4 text boxes. (Hint : Use of explode function.)
2. Write a PHP script for the following: Design a form to accept two strings. Compare the two strings using both methods (= = operator & strcmp function). Append second string to the first string. Accept the position from the user; from where the characters from the first string are reversed. (Use radio buttons)
3. Using regular expressions check for the validity of entered email-id. The @ symbol should not appear more than once. The dot (.) can appear at the most once before @ and at the most twice or at least once after @ symbol. The substring before @ should not begin with a digit or underscore or dot or @ or any other special character. (Use explode and ereg function.)

Signature of the instructor :_____ Date :_____

Assignment Evaluation

0:Not Done	☐	**2:Late Complete**	☐	**4:Complete**	☐
1:Incomplete	☐	**3:Needs Improvement**	☐	**5:Well Done**	☐

ASSIGNMENT NO. 2 : TO STUDY ARRAYS .

ARRAYS

An array is a collection of data values. Array is organized as an ordered collection of (key,value) pairs.

In PHP there are two kinds of arrays :

1. Indexed array : An array with a numeric index starting with 0.
   For example,
   Initializing an indexed array,
   $numbers[0]=100;
   $numbers[1]=200;
   $numbers[2]=300;

2. Associative array : An array which have strings as keys which are used to access the values.
   Initializing an Associative array,
   $numbers['one']=100;
   $numbers['two']=200;
   $numbers['three']=300;

Functions used with array :

Name	Use	Example
Array()	This construct is used to initialize an array.	$numbers=array(100,200,300); $cities=array('Capital of Nation'=>'Delhi', 'Capital of state'=>'Mumbai', 'My city'=>'Nashik');
List()	This function is used to copy values from array to the variables.	$cities=array('Capital of Nation'=>'Delhi', 'Capital of state'=>'Mumbai', 'My city'=>'Nashik'); list($cn,$cs,$c)=$cities; Output : $cn='Delhi' $cs='Mumbai' $c='Nashik'
Array_splice()	This function is used to remove or insert elements in array	$student=array(11,12,13,14,15,16); $new_student=array_splice($student,2,3); /* starting from index(2) and length =3 $new_student1=array_splice($student,2); /* here length is not mentioned */ Output : $new_student=(13,14,15); $new_student1=(13,14,15,16);
Array_key_exists()	This function is used to check if an element exist in the	$cities=array('Capital of Nation'=>'Delhi', 'Capital of state'=>'Mumbai', 'My city'=>'Nashik'); If (array_key_exists('Capital of State',$cities))

	array.	{
		echo “Key found!\n”;
		}
		Output :
		Key found!

Extract()	This function automatically creates local variables from the array.	extract($student); By this, the variables are created like this : $roll = 11; $name=’A’; $class=’TYBSc’;

Foreach()	This is the most common way to loop over elements of an array.
	PHP executes the body of the loop once for each element of $students, with $value set to the current element.
		For indexed array :
		
		$students=array(‘a’,’b’,’c’,’d’);
		foreach($student as $value)
		{
		echo “student $value \n”;
		}
		Output
		Student a
		Student b
		Student c
		Student d
		For associative array :
		$students=array(‘Name’=>‘a’,’Roll no’ => 100,
		‘class’=>’TYBSc’);
		foreach($student as $key=>$value)
		{
		echo “student’s $key is : $value \n”;
		}
		
		Student’s Name is : a
		Student’s Roll No is : 100
		Student’s class is : TYBSc

Array_push() Array_pop()	These functions are used to treat an array like a stack .	array_push($a); $b=array_pop();
Array_shift() Array_unshift()	These functions are used to treat an array like a queue.	array_shift(); array_unshift();

Set A

Q1) Write a menu driven program to perform the following operations on an associative array:

i. Display the elements of an array along with the keys.
ii. Display the size of an array
iii. **Delete an element from an array from the given index**.
iv. Reverse the order of each element's key-value pair [Hint: use array_flip()]
v. Traverse the elements in an array in random order [[Hint: use shuffle()].

Q:2) Accept a string from the user and check whether it is a palindrome or not (Implement stack operations using array built-in functions).

Set B

Q: 1) Declare a Multidimensional Array. Display specific element from a Multidimensional array. Also delete given element from the Multidimensional array.

Q: 2) Define an array. Find the element from the array that matches the given value using appropriate search function.

Set C

Q.1) Write a menu driven program to perform the following stack and queue related operations:[Hint: use Array_push(), Array_pop(), Array_shift(), array_unshift() functions]

i. Insert an element in stack
ii. Delete an element from stack
iii. Display the contents of stack
iv. Insert an element in queue
v. Delete an element from queue
vi. Display the contents of queue

Q: 2) Write a menu driven program to perform the following operations on associative arrays:

1. Sort the array by values (changing the keys) in ascending, descending order.
2. Also sort the array by values without changing the keys.
3. Filter the odd elements from an array.
4. Sort the different arrays at a glance using single function.
5. Merge the given arrays.
6. Find the intersection of two arrays.
7. Find the union of two arrays.
8. Find set difference of two arrays.

Signature of the instructor :_____ Date :_____

Assignment Evaluation

0:Not Done		**2:Late Complete**		**4:Complete**	
1:Incomplete		**3:Needs Improvement**		**5:Well Done**	

ASSIGNMENT NO. 3 : TO STUDY FILES AND DIRECTORIES.

File : A **file** is nothing more than an ordered sequence of bytes stored on hard disk, floppy disk CD-ROM or some other storage media.

Objectives :

Opening and closing a file
Reading a file and writing into file
Deleting and renaming a file
Navigating a file
Opening and closing directories
Reading directory entries
Deleting and renaming a directory

Note:-

- one differences between Linux and windows when it comes to specifying directory path is UNIX based system like LINUX use forward slash to delimit elements in a path
- A **file handle** is nothing more than an integer value that will be used to identify the file you wish to work with until it is closed

Working with files

<table>
<tr><th>Function Name</th><th>Description</th></tr>
<tr><td>fopen()</td><td>Opening and closing a file
This is used to open a file ,returning a file handle associated with opened file .It can take three arguments :fname,mode and optional use_include_path
Ex:-$fp=fopen(“data.txt”,r);
We can also open a file on remote host
List of modes used in fopen are:
<table>
<tr><th>Mode</th><th>Purpose</th></tr>
<tr><td>r</td><td>Open for reading only; place the file pointer at the beginning of the file</td></tr>
<tr><td>r+</td><td>Open for reading and writing; place the file pointer at the beginning of the file.</td></tr>
<tr><td>w</td><td>Open for writing only; place the file pointer at the beginning of the file and truncate the file to zero length. If the file does not exist, attempt to create it.</td></tr>
<tr><td>w+</td><td>Open for reading and writing; place the file pointer at the beginning of the file and truncate the file to zero length. If the file does not exist, attempt to create it.</td></tr>
<tr><td>a</td><td>Open for writing only; place the file pointer at the end of the file. If the file does not exist, attempt to create it.</td></tr>
<tr><td>a+</td><td>Open for reading and writing; place the file pointer at the end of the file. If the file does not exist, attempt to create it.</td></tr>
</table>
</td></tr>
</table>

fclose()	This is used to close file, using its associated file handle as a single argument Ex:- fclose(fp);
fread( )	This function is used to extract a character string from a file and takes two arguments, a file handle and a integer length Ex: fread($fp,10);
fwrite()	This function is used to write data to a file and takes two arguments, a file handle and a string Ex: fwrite($fp,"HELLO");
fgetc()	Function can be used to read one character from file at a fileIt takes a single argument ,a file handle and return just one character from the file .It returns false when it reached to end of file.
fgets()	This function is used to read set of characters it takes two arguments, file pointer and length. It will stop reading for any one of three reasons: • The specified number of bytes has been read • A new line is encountered • The end of file is reached
fputs()	This is simply an alias for fwrite() .
file()	This function will return entire contents of file. This function will automatically opens, reads, and closes the file. It has one argument : a string containing the name of the file. It can also fetch files on remote host.
fpassthru()	This function reads and prints the entire file to the web browser. This function takes one argument, file handle. If you read a couple of lines from a file before calling fpassthru(), hen this function only print the subsequent contents of a file.
readfile()	This function prints content of file without having a call to fopen() It takes a filename as its argument, reads a file and then write it to standard output returning the number of bytes read (or false upon error)
fseek()	It takes file handle and integer offset, offset type as arguments .It will move file position indicator associated with file pointer to a position determined by offset. By default this offset is measured in bytes from the beginning of the file. The third argument is optional, can be specified as: SEEK_SET:-Beginning of file +offset SEEK_CUR:-Current position +offset(default) SEEK_END:-End of the file +offset
ftell()	It takes file handle as an argument and returns the current offset (in bytes) of the corresponding file position indicator.
rewind()	It accepts a file handle as an argument and reset the corresponding file position indicator to the beginning of file.
file_exists()	It takes file name with detail path as an argument and returns

<table>
<tr><td></td><td>true if file is there otherwise it returns false</td></tr>
<tr><td>file_size()</td><td>It takes file name as an argument and returns total size of file (in bytes)</td></tr>
<tr><td>fileatime()</td><td>It takes filename as an argument and returns last access time for a file in a UNIX timestamp format</td></tr>
<tr><td>filectime()</td><td>It takes filename as an argument and returns the time at which the file was last changed as a UNIX timestamp format</td></tr>
<tr><td>filemtime()</td><td>It takes filename as an argument and returns the time at which the file was last modified as a UNIX timestamp format</td></tr>
<tr><td>fileowner()</td><td>It takes filename as an argument and returns the user ID of the owner of specified file.</td></tr>
<tr><td>posix_getpwuid()</td><td>It accept user id returned by fileowner() function as an argument and returns an associative array with following references
<table>
<tr><td>Key</td><td>Description</td></tr>
<tr><td>name</td><td>The shell account user name of the user</td></tr>
<tr><td>passwd</td><td>The encrypted user password</td></tr>
<tr><td>uid</td><td>The ID number of the user</td></tr>
<tr><td>Gid</td><td>The group ID of the user</td></tr>
<tr><td>Gecos</td><td>A comma separated list containing user full name office phone, office number and home phone number</td></tr>
<tr><td>Dir</td><td>The absolute path to the home directory of the user</td></tr>
<tr><td>Shell</td><td>The absolute path to the users default shell</td></tr>
</table></td></tr>
<tr><td>filegroup()</td><td>It takes filename as an argument and returns the group ID of owner of the specified file</td></tr>
<tr><td>posix_getgid()</td><td>It accept group ID returned by filegroup() function as an argument and returns an associative array on a group identified by group ID with following references
<table>
<tr><td>Key</td><td>Description</td></tr>
<tr><td>Name</td><td>The name of group</td></tr>
<tr><td>Gid</td><td>The ID number of group</td></tr>
<tr><td>members</td><td>The number of members belonging to the group</td></tr>
</table></td></tr>
<tr><td>filetype()</td><td>It takes filename as an argument and returns the type of specified file. the type of possible values are fifo, char, dir, block, link, file and unknown</td></tr>
<tr><td>basename()</td><td>It takes file name as an argument and separate the filename from its directory path.</td></tr>
<tr><td>copy()</td><td>It takes two string arguments referring to the source and destination file respectively.</td></tr>
<tr><td>rename()</td><td>It takes two argument as old name and new name and</td></tr>
</table>

	renames the file with new name.
unlink()	It takes a single argument referring to the name of file we want to delete.
is_file()	It returns true if the given file name refers to a regular file.

Examples

Use of some above mentioned functions is illustrated in the following examples:

Ex.1) To read file in server use **fread()** function. A file pointer can be created to the file and read the content by specifying the size of data to be collected.

```
<?php $body_content="This is my content"; // Store some text to enter in the file
$file_name="test_file.txt";              // file name
$fp = fopen ($file_name, "w");
// Open the file in write mode, if file does not exist then it will be created.
fwrite ($fp,$body_content);          // entering data to the file
fclose ($fp);                        // closing the file pointer
chmod($file_name,0777);              // changing the file permission.
?>
```

Ex.2) A file can be written by using **fwrite()** function in PHP. A file can be opened in write mode and if write permission is there then only it can be opened in write mode. If the file does not exist then one new file will be created. The file permissions can be changed.

```
<?php
$filecontent="Some text in File"; // Store some text to enter inside the file
$file_name="test_file.txt";              // file name
$fp = fopen ($filename, "w");  // Open the file in write mode, if it does not exist
then it will be created.
fwrite ($fp,$filecontent);          // entering data to the file
fclose ($fp);                        // closing the file pointer
chmod($filename,0777);              // changing the file permission.
?>
```

Ex.3) A small code for returning a **file-size.**

```
<?php
function dispfilesize($filesize){
   if(is_numeric($filesize)){
   $decr = 1024; $step = 0;
   $prefix = array('Byte','KB','MB','GB','TB','PB');
```

```
    while(($filesize / $decr) > 0.9){
       $filesize = $filesize / $decr;
       $step++;
    }
    return round($filesize,2).' '.$prefix[$step];
    } else {

    return 'NaN';
    }
}
?>
```

Ex.4) Here's a way to get a **files' extension**:

```
<?php
$file = $_FILES['userfile'];
$allowedExt = array("txt", "rtf", "doc");
function isAllowedExtension($fileName) {
  global $allowedExt;
  return in_array(end(explode(".", $fileName)), $allowedExt);
}
if($file['error'] == UPLOAD_ERR_OK) {
  if(isAllowedExtension($file['name'])) {
  } else {
    echo "Invalid file type";
  }} else die("Cannot upload");
?>
```

Working with Directories

A **directory** is special type of file that holds the names of other files and directories and pointer to their storage area on media. A **directory handle** is nothing more than an integer value pointing to a directory ,which can be obtained bf specifying the directory in call to the opendir() function.

Function Name	Purpose
opendir()	It takes directory name with detail path as an argument and returns directory handle on success, otherwise false.
closedir()	It takes directory handle as an argument and close directory
readdir()	It takes directory handle as an argument and returns the next entry listed in the open directory.
Other directory functions	
rewinddir()	It accepts a directory handle as an argument and reset the corresponding directory position indicator to the beginning of the directory

chdir()	This function changes current directory to given directory
rmdir()	It remove specified directory
mkdir()	It creates directory as specified in its first argument
dirname()	It returns directory part of given file name
Is_dir()	It returns true if the given file name refers to a directory.

Examples

Use of some of the above mentioned functions related to the directory is illustrated in the following examples:

Ex:1)

```
// open the current directory by opendir
$handle=opendir(".");

while (($file = readdir($handle))!==false) {
echo "$file <br>";
}

closedir($handle);
$path="../dir-name/";// path of the directory
$handle=opendir($path);
while (($file_name = readdir($handle))!==false) {
if(stristr($file_name,".php")) } // read the file
```

Set A:

1. A program to read two file names from user and append contents of first file into second file.

2. A program to read directory name from user and display content of the directory.

Set B:

1. Write a program to read a flat file student.dat and display the data from file in tabular format also calculate the percentage.

2. Write a program to read directory name and extension. Display the files with specified extension from that directory.

Set C:

1. Write a menu driven program to perform various file operations.

1. Display size of file
2. Display Last Access, changed, modified time of file
3. Display details about owner and user of File
4. Display type of file
5. Delete a file
6. Copy a file
7. Traverse a directory in hierarchy
8. Remove a directory

2. A program to read directory name from user and display content of the directory recursively.

Signature of the instructor :_____ Date :_____

Assignment Evaluation

0:Not Done		**2:Late Complete**		**4:Complete**	
1:Incomplete		**3:Needs Improvement**		**5:Well Done**	

ASSIGNMENT NO. 4 : OBJECT ORIENTED PROGRAMMING.

Objective : To understand Object oriented technique in PHP.

Class

A class is a unit of code composed of variables and functions which describes the characteristics and behavior of all the members of the set.

Function	Description	Example

Class classname [extends baseclass]	Creates a class	Class student
		{
		[var $propery [= value];…]
		[function functionname
		(arguments)
		{
		//code
		}
		….
		]
		}

$instance = new Classname();	Create an object	<?php $instance1 = new myclass (); //This can also be done with a variable: $newname= ‘hello’; $instance2 = new $newname(); ?>

Class classname
{
function methodname()
{
Statements;
}
}
	Add a Method	<?php
		Class myclass
		{ function mymethod()
		{ Print “ hello, myclass}}
		?>
		To invoke the method on the object $instance1, we need to invoke the operator -> to access the newly created function mymethod
		
		<?php
		$instance1=new myclass();
		$instance1->mymethod();
		?>

public $publicMemeber = “Public member”;	Adding Property
	
	Public
		Public :
		<?php
		class maths
		{
		public $num;
		public function multi()
		{
		return $this->num*2;
		}
		}
		$math=new maths;
		$math->num=2;
		echo $math->multi();
		?>
		
		Output : 4

protected $protectedmember = “Protected Member”;
Private $privatemember= “Private Member
	Protected
	Private
		Protected:
		
		<?php
		class maths
		{
		protected $num;
		public function setnum($num)
		{
		$this->num=$num;
		}
		
		public function multi()
		{
		return $this->num*2;}}
		
		class add extends maths
		{
		public function addtwo()
		{
		$new_num=$this->num + 2;
		return ($new_num);
		}
		}
		$math=new add;
		$math->setnum(14);
		echo $math->addtwo();
		?>
		
		Output : 16

class extendedClass
extends classname
	Inheritance
	It is the ability of PHP to extend classes that inherit the characteristics of the parent class.
	
	It is not possible to extend multiple classes ; a class can only inherit from one base class.
		<?php
		class myclass
		{
		//property declaration
		public $var='a default value';
		
		//method declaration
		public function desplayVar()
		{
		echo $this->var;
		}
		}
		
		class extendedClass extends myclass
		{
		//redefine the parent method
		function displayVar()
		{
		echo "Extending Class";
		parent::displayVar();
		}
		}
		
		$extend =new extendedClass();
		$extend->displayVar();
		?>
		
		Output :
		Extending class
		a default value

Overriding	When we give a function in the child class the same name as a function in the parent class, this concept is called function overriding.
	
	Any method or class that is declared as final can not be overridden or inherited by another class.
		<?php
		class Hello
		{function getMessage()
		{ return 'Hello World !';}
		}
		class Goodbye extends Hello
		{function getMessage(){
		return 'Goodbye World!';}}
		$hello=&new Hello();
		Echo $hello->getMessage().'
		';
		$goodbye = &new Goodbye();
		Echo $goodbye->getMessage().
		'
		';?>
		Output:
		Hello World!
		Goodbye World!

void _construct ([mixed $args [, $....]])	Constructor is a function which is called right after a new object is created.	<?php
		class Student
		{
		var $name;
		var $address;
		var $phone;
		
		//This is constructor
		function student()
		{
		this->name=”abc”;
		this->address=”pqr”;
		this->phone=1111;
		}
		
		function printstudentinfo()
		{
		echo this->name . ”\n”;
		echo this->address . ”\n”;
		echo this->phone . “\n”;
		}
		}
		$stud =new student();
		$stud->printstudentinfo();
		$stud=NULL;
		?>

void _destruct (void)	Destructor is a function which is called right after you release an object.	<?php
		class Student
		{
		var $name;
		var $address;
		var $phone;
		
		//This is constructor
		function _construct()
		{
		this->name=”abc”;
		this->address=”pqr”;
		this->phone=1111;
		}
		
		function _destruct()
		{
		echo “Student Object Released”;}

		function printstudentinfo()
		{
		Echo this->name . ”\n”;
		echo this->address . ”\n”;
		echo this->phone . “\n”;
		}
		}
		$stud =new student();
		$stud->printstudentinfo();
		$stud=NULL;
		?>

class_exist() get_declared_classes() get_class_methods() get_class_vars() get_parent_class()	**Introspection** We can use this function to determine whether a class exists. This function returns array of defined classes and checks if the class name is in returned array. We can use this function to get the methods and properties of class This function returns only properties that have default values. This function is used to find the class’s parent class.	$class = class_exists(classname); $classes = get_declared_classes(); $methods = get_class_methods(classname); $properties=get_class_vars(classname); $superclass=get_parent_class(classname);
is_object() get_class()	Is_object function is used to make sure that it is object. get_class() function is used to get the class to which an object belongs and to get class name This function is used to check if method on an	$obj= is_obj(var); $classname= get_class(object);

	object exists.	
method_exists()	This function returns an array of properties set in an object	$method_exists=method_exists(object ,method);
get_object_vars()	This function returns the properties of object that have values assigned to them.	$array=get_object_vars(object);
serialize()	**Serialization** Serializing an object means converting it to a byte stream representation that can be stored in a file. returns a string containing a byte-stream representation of the value that can be stored anywhere	$encode=serialize(something)
unserialize()	Takes a single serialized variable and converts it back to PHP value.	$something = unserialize(encode);

Interfaces	An interface is declared similar to a class but only include function prototypes (without implementation) and constants.
	When a class uses an interface the class must define all the methods/function of the interface otherwise the PHP engine will give you an error.
	
	The interface's function /methods cannot have the details filled in. that is left to the class that uses the interface.
		Example of an interface
		class duck
		{ function quack()
		{
		echo "quack,quack,qk, qk…";
		}
		}
		Interface birds
		{
		function breath();
		function eat();
		}
		Class duck implements birds
		{ function quack()
		{
		echo "quack,quack,qk, qk…";
		}
		}
		function breath()
		{
		echo "duck is breathing";
		}

		function eat()
		{
		echo " duck is eating";
		}

Encapsulation	Encapsulation is an ability to hide details of implementation.	<?php
		class A
		{
		function check()
		{
		if(isset ($this))
		{
		echo "$this is defined (";
		echo get_class($this);
		echo ")\n";
		}
		else
		{
		echo "this is not defined";
		}
		}
		}
		class B
		{
		function bcheck()
		{
		A::check();
		}
		}
		
		$a=new A();
		$a->check();
		A::check();
		$b=new B();
		$b->bcheck();
		B::bcheck();
		?>
		
		Output:
		
		$this is defined(a)
		$this is not defined
		$this is defined(b)
		$this is not defined

Set A

Q: 1) Define an interface which has methods area(), volume(). Define constant PI. Create a class cylinder which implements this interface and calculate area and volume. (Hint: Use define())

Q: 2) Write class declarations and member function definitions for an employee(code, name, designation). Design derived classes as emp_account(account_no, joining_date) from employee and emp_sal(basic_pay, earnings, deduction) from emp_account.

Write a menu driven program

a) to build a master table
b) to sort all entries
c) to search an entry
d) Display salary.

Set B

Q:1) Derive a class square from class Rectangle. Create one more class circle. Create an interface with only one method called area(). Implement this interface in all the classes. Include appropriate data members and constructors in all classes. Write a program to accept details of a square, circle and rectangle and display the area.

Q:2) Create a class account(accno,cust_name). Derive two classes from account as saving_acc(balance, min_amount) and current_acc(balance, min_amount).

Display a menu

a) Saving Account
b) Current Account

For each of this display a menu with the following options.

1. Create account
2. Deposit
3. Withdrawal

Set C

Q:1) Define an interface for stack operation. Implement this interface in a class.

Q:2) Write necessary class and member function definitions for a cricket player object.The program should accept details from user (max :10) (player_code, name, runs, innings_played, no_of_times_out).

The program should contain following menu.

a) Enter details of players.
b) Display average runs of a single player.
c) Average runs of all players.
d) Display the list of players in sorted order as per runs(use function overloading)

Signature of the instructor :_____ Date :_____

Assignment Evaluation

0:Not Done		**2:Late Complete**		**4:Complete**	
1:Incomplete		**3:Needs Improvement**		**5:Well Done**	

ASSIGNMENT NO. 5 : COOKIES & SESSIONS.

Objective : To demonstrate use of cookies and sessions.
Reading : You should read the following topics before starting this exercise.

1. Introduction to cookies.
2. Introduction to session.

- A **cookie** is a small amount of data stored by the user's browser in compliance with a request from a server or script.
- **Session** allow us to easily create multi page forms, save user authentication information from page to page, and store persistent user preferences on a site.
  A session can be defined as a series pf related interactions between a single client and the Web server. The session may consist of multiple requests to the same script or a variety of different resources on the same web site.

Action Involving Cookies	Syntax	Example

Create cookie	setcookie(name [, value [, expire [, path [, domain [, secure]]]]]);
	
	name – A unique name for a particular cookie.
	Value – string value attached to this cookie.
	Expire – expiration date is specified as no of second since midnight Jan, 1, 1970 GMT.
	Path – the browser will return the cookie only for URL, below this path.
	Domain – the browser will return the cookie only for URLs within this domain.
	Secure – the default is false.
		<?php
		setcookie("item", "TV", time()+3600, "/", ".yourdomain.com", 0);
		if(isset($_COOKIE["item"]))
		{
		echo "Hello again, you have chosen : " . $_COOKIE["item"];
		}
		else
		{
		echo "Hello you. This may be your first visit.";
		}
		?>

Delete cookie	set the cookie with a date that you are sure has already expired.	setcookie("item", "", time()-60, "/", ".yourdomain.com", 0);
Create session	**session_start()** – to enable session for a page. This function assigns a new session ID to the new session	<?php session_start(); ?> <?php session_start();

	session_register() – to register a variable with the session by passing the name of the variable. When a session is started, you can store any number of variables in the $_SESSION superglobal array and then access them on any session-enabled page. **session_id( )** function returns the current session ID.	session_register('hits'); ++$hits; ?> This page has been viewed <?= $hits ?> times.
End session	**session_destroy( )** This function removes the data store for the current session without removing cookie from the browser cache. **session_unregister(name)** unregisters the global variable named *name* from the current session	

Cookies are very powerful and reliable method of storing small piece of data needed to be stored on client's machine but not permanently between separate visits to a Web site.

- A Cookie is a small part of data that can be used to store a variable's name, its value along with the information on the site from which it came and its expiry time.
- Cookies provide client-side storage in files present on client machine's hard drive.
- Cookies can be accessed and changed by the web server from which they were sent originally.

Set A

Q:1) Create a login form with a username and password. Once the user logs in, the second form should be displayed to accept user details (name, city, phoneno). If the user doesn't enter information within a specified time limit, expire his session and give a warning.

Q:2) Write a script to keep track of number of times the web page has been accessed.

Set B

Q:1) Change the preferences of your web page like font style, font size, font

color, background color using cookie. Display selected settings on next web page and actual implementation (with new settings) on third web page.

Q:2) Create a form to accept student information (name, class, address). Once the student information is accepted, accept marks in next form (Phy, Bio, Chem, Maths, Marathi, English) .Display the mark sheet for the student in the next form containing name, class, marks of the subject, total and percentage.

Set C

Q:1) Write a program to create a shopping mall. User must be allowed to do purchase from two pages. Each page should have a page total. The third page should display a bill, which consists of a page total of what ever the purchase has been done and print the total. (Use http session tracking).

Q:2) Create a form to accept customer information(name, address, ph-no).Once the customer information is accepted, accept product information in the next form(Product name, qty, rate). Display the bill for the customer in the next form. Bill should contain the customer information and the information of the products entered.

Q:3) Write a PHP script to accept username and password . If in the first three chances, username and password entered is correct, then display second form, otherwise display error message.

Signature of the instructor :_____ Date :_____

Assignment Evaluation

0:Not Done		**2:Late Complete**		**4:Complete**	
1:Incomplete		**3:Needs Improvement**		**5:Well Done**	