CMR
INSTITUTE OF
TECHNOLOGY

USN | | | | | | | | | | |

Internal Assessment Test 1 Answer Key– April. 2021									
Sub:	Advanced Web Programming					Sub Code:	18MCA42	Branch:	MCA
Date:	06/04/2021	Duration:	90 min's	Max Marks:	50	Sem	IV		

Q1) Explain Cookies in PHP with example (10 marks)

- A cookie is a small object of information that consists of a name and a textual value. A cookie is created by some software system on the server.
- The header part of an HTTP communication can include cookies. So, every request sent from a browser to a server, and every response from a server to a browser, can include one or more cookies.
- At the time it is created, a cookie is assigned a lifetime. When the time a cookie has existed reaches its associated lifetime, the cookie is deleted from the browser's host machine.
- Cookie is set in PHP with setcookie function
- First parameter is cookie's name given as a string. The second, if present, is the new value for the cookie, also a string. If the value is absent, setcookie undefines the cookie.
- The third parameter, when present, is the expiration time in seconds for the cookie, given as an integer.
- The default value for the expiration time is zero, which specifies that the cookie is destroyed at the end of the current session. When specified, the expiration time is often given as the number of seconds in the UNIX epoch, which began on January 1, 1970. The time function returns the current time in seconds. So, the cookie expiration time is given as the value returned from time plus some number.
- For example,
  setcookie("voted", "true", time() + 86400);

  This call creates a cookie named "voted" whose value is "true" and whose lifetime is one day (86,400 is the number of seconds in a day).
- To delete a cookie, use the setcookie() function with an expiration date in the past:

  setcookie("voted", "true", time() - 86400);

- All cookies that arrive with a request are placed in the implicit $ COOKIES array, which has the cookie names as keys and the cookie values as values.
- We can retrieve the value of the cookie using the global variable $_COOKIE

  $_COOKIE[$cookie_name]
  Eg. $_COOKIE["voted"]

Q2) Explain Session tracking in PHP with example (10 marks)

In many cases, information about a session is needed only during the session. Also, the needed information about a client is nothing more than a unique identifier for the session, which is commonly used in shopping cart applications. For these cases, a different process, named session tracking, can be used.

- Rather than using one or more cookies, a single session array can be used to store information about the previous requests of a client during a session.
- In particular, session arrays often store a unique session ID for a session.
- One significant way that session arrays differ from cookies is that they can be stored on the server, whereas cookies are stored on the client.

- In PHP, a session ID is an internal value that identifies a session. Session IDs need not be known or handled in any way by PHP scripts.
- PHP is made aware that a script is interested in session tracking by calling the session start function, which takes no parameters. The first call to session start in a session causes a session ID to be created and recorded.
- On subsequent calls to session_start in the same session, the function retrieves the $_SESSION array, which stores any session variables and their values that were registered in previously executed scripts in this session.
- Session key/value pairs are created or changed by assignments to the $ SESSION array.
- They can be destroyed with the unset operator.
- Consider the following example: session_start();
  if (!IsSet($_SESSION["page_number"]))
  $ SESSIONppage_number") = 1;
  $page_num = $_SESSION["page_number"];
  print("You have now visited $page_num page(s)
");
  $ SESSIONppage number")++;
- If this is not the first document visited that calls session start and sets the page_number session variable, this script will produce the specified line with the last set value of $_SESSION ["page_number") .
- If no document that was previously visited in this session set page_number, this script sets page_number to 1, produces the line 'You have now visited 1 page(s)', and increments page_number

Q3) Explain with examples String pre-defined functions in PHP (10 marks)

Function Name	Parameters	Description
strlen	One string	Returns number of characters in the string
strcmp	Two strings	Returns zero if both strings are equal, a -ve number if the first string occurs before second string or a +ve number if the first string occurs after the second string
strpos	Two strings	Returns position of second string in the first string or false if not found
substr	One string and one integer	Returns the substring from the specified string from the position specified as an integer. If a third integer value is specified, it represents the length of the substring to be retrieved
chop	One string	Returns the string with all white space characters removed from the end
trim	One string	Returns the string with all white space characters removed on both sides
ltrim	One string	Returns the string with all white space characters removed from the beginning
strtolower	One string	Returns the string with all the characters converted to lowercase
strtoupper	One string	Returns the string with all the characters converted to uppercase
strrev	One string	Returns the reverse of the given string
str_replace	Three strings	Returns the string in which a old substring is replaced by the new substring
str_word_count	One string	Returns the word count in the given string

Q4) Explain with examples numeric pre-defined functions in PHP (10 marks)

Table 11.2 Some useful predefined functions

Function	Parameter Type	Returns
`floor`	Double	Largest integer less than or equal to the parameter
`ceil`	Double	Smallest integer greater than or equal to the parameter
`round`	Double	Nearest integer
`srand`	Integer	Initializes a random number generator with the parameter
`rand`	Two numbers	A pseudorandom number greater than the first parameter and smaller than the second
`abs`	Number	Absolute value of the parameter
`min`	One or more numbers	Smallest
`max`	One or more numbers	Largest

Q5) Explain Functions in php with example (10 marks)

A function is a part of program which contains set of statements that can perform a desired task. A function can be defined with the *function* keyword followed by the function name and an optional set of parameters enclosed in parentheses. A function definition may not occur before a function call. It can be anywhere in the script. Although function definitions can be nested, it is not encouraged as they can make the script complex. Syntax for defining a function is given below:

```
function  function_name([param1, param2, ....])
{
        //Body of the function
        [return expression;]
}
```

The execution of the function terminates whenever a *return* statement is encountered or whenever the control reaches the end of the function's body.

Parameters

As in any typical programming language, parameters in a function definition are known as *formal parameters* and the parameters in a function call are known as *actual parameters*. The number of formal parameters and normal parameters may not be equal.

The default parameter passing mechanism is pass-by-value where a copy of the actual parameters are passed to actual parameters. Below example demonstrates pass-by-value:

```
function swap($x, $y)
{
        $temp = $x;
        $x = $y;
        $y = $temp;
}
$a = 10;
$b = 20;
sum($a, $b);
```

After the above code is executed the values of *a* and *b* remains 10 and 20 even after the function call.

Another way of passing parameters is *pass-by-reference* where the address of the variable is passed rather than a copy. In this mechanism changes on formal parameters are reflected on actual parameters. The address can be passed using & symbol before the variable as shown below:

```
function swap($x, $y)
{
        $temp = $x;
        $x = $y;
        $y = $temp;
}
$a = 10;
$b = 20;
sum(&$a, &$b);
```

or the above code may be also written as shown below:

```
function swap(&$x, &$y)
{
        $temp = $x;
        $x = $y;
        $y = $temp;
}
$a = 10;
$b = 20;
sum($a, $b);
```

Pass-by-reference can be used to return multiple values from a function.

Scope of Variables

Scope refers to the visibility of a variable in the PHP script. A variable defined inside a function will have local scope i.e., within the function. A variable outside the function can have the same name as a local variable in a function, where the local variable has higher precedence over the outer variable.

In some cases, a function might need access to a variable declared outside the function definition. In such cases, *global* declaration can be used before the variable name which allows the accessibility of the variable that is declared outside the function definition. Below example demonstrates local and global variables:

```
function sum_array($list)
{
        global $allsum; //Global variable
        $sum = 0; //Local variable
        foreach($list as $x)
                $sum += $x;
        $allsum += $sum;
        return $sum;
}
$allsum = 0;
$array1 = new array(10, 20, 30, 40);
$array2 = new array(1, 2, 3, 4);
$ans1 = sum_array($array1);
$ans2 = sum_array($array2);
print("Sum of array1 is: $ans1<br />");
print("Sum of all arrays is: $allsum");
```

In the above example, first print statement gives 100 and the second print statement gives 110.

Lifetime of Variables

The lifetime of a normal variable in a function is until the function execution completes. Sometimes, there might be a need to retain the value of a variable between function calls. In such cases, the variable must be declared with *static*. The lifetime of *static* variable is until the execution of script completes. Below example demonstrates the use of *static* variables:

```
function func1( )
{
        static $count = 0;
        $count++;
}
for($i = 1; $i <= 10; $i++)
        func1( );
print("func1 is called $count times.");
```

In the above example, the *print* statement prints 10 instead of 0.

Q6) How to create array in PHP? Explain different functions for dealing with array (10 marks)

Arrays

Array is a collection of heterogeneous elements. There are two types of arrays in PHP. First type of array is a normal one that contains integer keys (indexes) which can be found in any typical programming languages. The second type of arrays is an *associative array*, where the keys are strings. Associative arrays are also known as hashes.

Array Creation

Arrays in PHP are dynamic. There is no need to specify the size of an array. A normal array can be created by using the integer index and the assignment operator as shown below:

```
$array1[0] = 10;
```

If no integer index is specified, the index will be set to 1 larger than the previous largest index value used. Consider the following example:

```
$array2[3] = 5;
$array2[ ] = 90;
```

In the above example, 90 will be stored at index location 4.

There is another way for creating an array, using the *array* construct, which is not a function. The data elements are passed to the *array* construct as shown in the below example:

```
$array3 = array(10, 15, 34, 56);
$array4 = array( );
```

In the above example *array4* is an empty array which can be used later to store elements.

A traditional array with irregular indexes can be created as shown below:

```
$array5 = array(3 => 15, 4 =>, 37, 5 => 23);
```

The above code creates an array with indexes 3, 4 and 5.

An associative array which contains named keys (indexes) can be created as shown below:

```
$ages = array("Ken" => 29, "John" => 30, "Steve" => 26, "Bob" => 28);
```

An array in PHP can be a mixture of both traditional and associative arrays.

Array Functions

The array elements can be manipulated or accessed in different ways. PHP has an extensive list of predefined functions that comes in handy while working with arrays. Some these functions are mentioned below:

Function Name	Parameters	Description
count	One array	Returns the number of elements in the array
unset	One array	Deletes the element from memory
array_keys	One array	Returns the keys (indexes) as an array
array_values	One array	Returns the values as an array
array_key_exists	Key, array	Returns *true* if the key is found, *false* otherwise
is_array	One array	Returns *true* if the argument is an array, *false* otherwise
in_array	Exp, array	Returns *true* if *exp* is in the array, *false* otherwise
explode	Delim, string	Returns an array of substrings based on the *delim*
implode	Delim, array	Returns a string joining the array elements with *delim*
sort	One array	Sorts the values in the array and renames the keys to integer values 0, 1, 2, ...
asort	One array	Sorts the values in the array preserving the keys
ksort	One array	Sorts the keys in the array preserving the values
rsort	One array	Reverse of sort (descending order)
rasort	One array	Reverse of asort (descending order)
rksort	One array	Reverse of ksort (descending order)

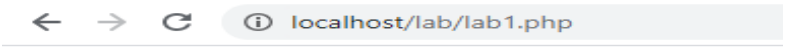
Q7) Write a PHP program to process a file which contains English words. Where each word is separated from the next word on a line by one space. The file is specified on the command line. The output of your program is a table in which the first column has unique words from the input file and the second column has the number of times the word appeared in the file; no word can appear twice in the table. Use two arrays to store the table, on for the words and one for the frequency values. (10 marks)

```
<?php
        function splitter($str){
                $freq=array();
        $words=preg_split('/\s+/',$str);
                foreach ($words as $word){
                        $keys=array_keys($freq);
                        if(in_array($word,$keys))
                                $freq[$word]++;
                        else
                                $freq[$word]=1;
                }
                return $freq;
        }
        $str = file_get_contents("words.txt","r");
        $tbl=splitter($str);
        print "<br/> Word Frequency<br/><br/>";
        $sorted_keys=array_keys($tbl);
        sort($sorted_keys);
```

```
        foreach($sorted_keys as $word)
                print "$word $tbl[$word]<br/>";

?>
```

OUTPUT



Word Frequency

apples 3
are 1
better 1
don't 1
for 1
good 1
like 2
maybe 1
or 2
oranges 1
than 1
you 3

Q8) A file contains lines of employee data, where each line has name:age:department code:salary. Write a PHP program to generate the following output:
i)The names of all the employee whose names end with "son"
ii)Percentage of employee under 40 years old (10 marks)

```
<?php
   $file = fopen("employees.txt","r");
    $under_40=0;
    $salary_sum=0;
    $emplist=[];
    echo "Names of all employees whose names end with son : <br/>";
    while(! feof($file))
     {
            $total_employees++;
            $fline=fgets($file);
            list($name,$age,$dept,$salary)=split(':',$fline);

            if(preg_match("/son$/",$name)){
                    echo $name."<br/>";
            }

            if($age<40){
                    $under_40++;


            }
     }

    if($total_employees>0){
            if($under_40>0){
                            $percent=100*$under_40 / $total_employees;
                            echo "<br/>Percent of employees under 40 is :".$percent."%<br/>";

\$40000 <br/>";
```

```
                }
                else{
                        echo"There are no employees under 40 <br/>";
                }
        }
        else{
                echo"There are no employees ";
        }
        fclose($file);
?>
```

Q9) Write an XHTML document to create a form with the following capabilities: text widget to collect the user's name four checkboxes, one each for the following items.
i)Four 100-watt light bulb for $2.39
ii)Eight 100- watt light bulbs for $4.29
iii)Four 100- watt long life light bulbs for $3.95
iv)Eight 100- watt long-life light bulbs for $7.49
A collection of three radio buttons they are labeled as follows:
i)Visa
ii)MasterCard
iii)Rupay
(10 marks)

lab4.xhtml

```
<?xml version = "1.0" encoding = "utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <title>Program 4</title>
    </head>
    <body>
        <h2>Bulbs Sales</h2>
        <form method="post" action="bulbs.php">
            <table>
                <tr>
                    <td>Buyer's Name:</td>
                    <td><input type="text" name="uname" size="30"/></td>
                </tr>

            </table>
            <br/>
            <table border="1">
                <tr>
                    <th>Product Name</th>
                    <th>Price</th>
                    <th>Quantity</th>
                </tr>
                <tr>
                    <td>Four 100-watt light bulbs</td>
                    <td>$2.39</td>
                    <td><input type="checkbox" name=" order []" value="2.39"
/></td>
```

```
                        </tr>
                        <tr>
                                <td>Eight 100-watt light bulbs</td>
                                <td>$4.29</td>
                                <td><input type="checkbox" name=" order []" value="4.29"
/></td>
                        </tr>
                        <tr>
                                <td>Four 100-watt long-life light bulbs</td>
                                <td>$3.95</td>
                                <td><input type="checkbox" name=" order []" value="3.95"
/></td>
                        </tr>
                        <tr>
                                <td>Eight 100-watt long-life light bulbs</td>
                                <td>$7.49</td>
                                <td><input type="checkbox" name=" order []" value="7.49"
/></td>
                        </tr>
                </table>
                <br/>
                <h3>Payment Method:</h3>
                <p>
                        <label><input type="radio" name="payment" value="Visa"
checked="checked"/>Visa</label>
                        <label><input type="radio" name="payment"
value="MasterCard"/>Master Card</label>
                        <label><input type="radio" name="payment"
value="Rupay"/>Rupay</label>
                </p>
                <p>
                        <input type="submit" value="Submit Order" />
                        <input type="reset" value="Clear Order Form" />
                </p>
        </form>
    </body>
</html>
```

bulbs.php

```
<?php
$name=$_POST ["uname"];
$bulbs=$_ POST ["order "];
$pmode=$_ POST ["payment"];
echo"<h3>Order Details</h3>";
print "Name: $name <br/>";
$total=0;
print "<br/><u>You have orderd following items:</u><br/>";
for($i=0;$i<sizeof($bulbs);$i++)
{
   $total=$total+$bulbs[$i];
    if($bulbs[$i]==2.39){
            echo"Four 100-watt light bulbs for $2.39<br/>";
    }
    elseif($bulbs[$i]==4.29){
            echo"Eight 100-watt light bulbs for $4.29<br/>";
```

```
        }
        elseif($bulbs[$i]==3.95){
                echo"Four 100-watt long-life light bulbs for $3.95<br/>";
        }
        else{
                echo"Eight 100-watt long-life light bulbs for $7.49<br/>";
        }
}

print "<br/>The total cost is : $total<br/>";
print "<br/>Your payment mode is : $pmode";
?>
```

Q10) Write a PHP program to insert name and age information entered by the user into a table created using MySQL and to display the current contents of this table. (10 marks)

5.xhtml

```
<html>
<body>
<form action=" 5.php" method="POST">
<table align="center" width="100" height="100">
<caption> Student Information</caption>
<tr><td>Name:</td><td><input type="text" name="name"></td></tr>
<tr><td>Age :</td><td><input type="text" name="age"></td></tr>
<tr><td></td><td><input type="submit" value="Submit"></td></tr></table></form>
</body>
</html>
```

5.php

```
<?php
$name=$_POST['name'];
$age=$_POST['age'];
mysql_connect("localhost","root","");
mysql_select_db("web1");
mysql_query("insert into p3(name,age) values('$name','$age')");

$result=mysql_query("select * from p3 where name='$name' and age='$age'");
$row=mysql_fetch_row($result);

?>
<table>
<tr>
<th>Name</th>
<th>age</th>
</tr>
<tr>
<td><?php echo $row[0];?></td>
<td><?php echo $row[1];?></td>
</tr>
</table>
```