Progressive Education Society's Modern College of Arts, Science and Commerce (Autonomous) Shivajinagar, Pune – 5

S Y B.C.A (SEM-IV)

Lab II: Web Technologies -I Lab Book

Course Code: 19ScBCAU406

Name: _____

RollNo.: _____

AcademicYear:_____

Introduction

Please read the following instructions carefully and follow them.

Students are expected to carry this book every time when they come to the lab for computer science practical.

1. Students should prepare themselves beforehand for the Assignment by reading the relevant material.
2. Instructor will specify which problems to solve in the lab during the allotted slot and student should complete them and get verified by the instructor. However student should spend additional hours in Lab and at home to cover as many problems as possible given in this work book.
3. Students will be assessed for each exercise on a scale from 0 to 5.

Not done	0
Incomplete	1
Late Complete	2
Needs improvement	3
Complete	4
Well Done	5

Guidelines for Lab Administrator

You have to ensure appropriate hardware and software is available to each student in the Lab.
The operating system and software requirements on server side and also client side are as given below:

1) Server and Client Side - (Operating System) Linux (Ubuntu/Red Hat/Fedora) – any distribution, IIS server
2) Database server – PostgreSQL 7.0 onwards.

Assignment Completion Sheet

Subject: Web Technologies - I			
Sr. No.	**Assignment Name**	**Marks (Out of 5)**	**Teacher's Sign**
1	PHP Programming		
2	Use of Functions and String		
3	Use of Arrays		
4	Object oriented programming		
5	Accessing Databases (PostgreSQL)		
Total Out of 40			

CERTIFICATE

This is to certify that

*Mr./Ms.*_____

has successfully completed Web technologies -I Laboratory course in the year _____ and his/her seat number is _____. He/She has scored mark _____ out of 40.

Internal Examiner **External Examiner** **H.O.D.**

Assignment Number 1: PHP Programming

Aim: To study simple programmes in PHP using flow control statements.

Pre-requisite:

- Knowledge of C/CPP programming language

Guidelines for Teachers/Instructors:

- Demonstration of variable declaration and loop syntax

Instructions for Students:

· Students must read theory and syntax for solving programmes before his/her practical slot.
· Solve SET A, B or C assigned by instructor in allocated slots only.

Theory:

PHP is server-side scripting language which is used to create dynamic web page. The long form of PHP is "Hypertext Pre-processor". Which is recursive. A script is a set of programming instructions that is interpreted at runtime. A scripting language that interprets scripts at runtime. The purpose of the scripts is usually to enhance the performance or perform routine tasks for an application. PHP is a server-side scripts that is interpreted in the server.

Reading:

https://www.tutorialspoint.com/php/php_tutorial.pdf

https://sites.harding.edu/fmccown/php_introduction.pdf

https://www.studytonight.com/php/introduction-to-php

Creating (Declaring) PHP Variables

In PHP, a variable starts with the $ sign, followed by the name of the variable: **Example**

```
<?php
$txt = "Hello world!";
$x = 5;
$y = 10.5;
?>
```

After the execution of the statements above, the variable $txt will hold the value Hello world!, the variable $x will hold the value 5, and the variable $y will hold the value 10.5.
A variable can have a short name (like x and y) or a more descriptive name (age, carname, total_volume).

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: **Example**

```
<?php
$txt = "PHP Programming.";
echo "I love $txt!";
?>
```

Flow Control Statements

The control statements are used to control the flow of execution of the program. This execution order depends on the supplied data values and the conditional logic. There are two conditional statement if..else and switch.

if ..else statement:

syntax:

```
if(condition1)
{//code block of if branch to be executed.
}elseif(codition2)
{//code block of elseif branch to be executed
}else
{//code block of else branch to be executed.
}
```

Switch statement:

```
switch(variable){

case value 1: //code block 1;
break;
case value 2: //code block 2;
break;
case value 3: //code block 3;
break;
……
```

default: //default code block;

}

Loops:

Loops are used to execute same block of statements specific number of times. There are 4 types of loops in PHP.

1) While loop: while loop will execute block of code until certain condition is meet.

Syntax:

while(expression)

{// code block to be executed

}

Example

```
<?php
$i=1;
while($i=10)
{ echo($i<=10)
{
echo $i.” “;
$i++;
}
?>
```

Output: 1 2 3 4 5 6 7 8 9 10

2) do..while: This loop works same as while loop , except that the expression is evaluated at the end of each iteration.

Syntax:

do {

//code block to be executed

}while(expression);

example

```
<?php
$i=1;
do
{ echo($i<=10)
{
echo $i.” “;
$i++;
} while($i=10);
```

```
?>
```

Output:

1 2 3 4 5 6 7 8 9 10

3) for: The for loop is used when you know in advance how many times the script should run.

Syntax

```
for (init counter; test counter; increment counter) {
 code to be executed for each iteration;
}
```

Parameters:

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

Example

The example below displays the numbers from 0 to 10

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

4) foreach Loop: The foreach loop - Loops through a block of code for each element in an array.

Syntax:

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

For every loop iteration, the value of the current array element is assigned to $value and the array pointer is moved by one, until it reaches the last array element.

Example

```
<?php
$colors = array("red", "green", "blue", "yellow");

foreach ($colors as $value) {
 echo "$value <br>";
}
?>
```

Exercises

SET A

1. Write a PHP script to get the PHP version and configuration information
2. Write a PHP script to display student information on web page.
3. Write a PHP script to script to display time table of your class (use HTML table tags in echo).

SET B

1. Write a PHP script to declare three variables and print maximum among them.
2. Write a PHP script to check number 153 is Armstrong or not.
3. Write a PHP script to check whether accepted number is prime or not.

SET C

1. Write a PHP script to print following floyd's triangle.

   1
   2 3
   4 5 6
   7 8 9 10
2. Write a PHP script to display source code of a webpage.
3. Write a PHP script to test whether a number is greater than 30, 20 or 10 using ternary operator.

Assignment Evaluation

0: Not Done [ ] 1: Incomplete [] 2: Late Complete []

3: Needs Improvement [] 4: Complete [] 5: Well-done []

Signature of Instructor

Date:

Assignment Number 2: Functions

Aim: To study user defined and library functions and their implementation in programs.

Pre-requisite:

· Knowledge of flow control statements, variable declaration
· Concept of string, HTML Tags

Instructions for Teachers/Instructors:

· Demonstration of user defined and library functions with their syntax using simple programmes

Instructions for Students:

· Students must read theory and syntax for solving programmes before his/her practical slot.
· Solve SET A, B or C assigned by instructor in allocated slots only.

Theory:

PHP is web scripting language implies that PHP is used to write web scripts, not the stand-alone applications. That is programs are executed through a web browser. PHP scripts run only after an event occurs.

Example hello.php

<html> <head> <title>Look Out World</title> </head> <body> <?php echo 'Hello, world!' ?> </body> </html>	Output: Hello, world!

User-defined 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. 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.

Example:

```
<?php
function Message () {
  echo "PHP";
}

Message (); // call the function
?>
```

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.

```
<?php

function display($greeting, $message=”GoodDay”)
{ echo $greeting;

        echo–<br>ǁ;
        echo $message;
}
display(Helloǁ);
?>
```

Output:

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

string of characters is probably the most commonly used data type when developing scripts, and PHP provides a large library of string functions to help transform, manipulate, and otherwise manage strings.

Function Name	**Description**	**Syntax**	**Example**
Strtolower()	The strtolower(s tring)	string strtolower (string $string)	<?php $str="India is great"; $str=strtolower($str); echo $str; ?> india is great
strtoupper()	The strtoupper(fun returns)	string strtoupper (string $string)	<?php $str="India is great"; $str=strtoupper($str); echo $str; ?> INDIA IS GREAT
ucfirst()	The ucfirst() function returns string converting first character into uppercase. It doesn't change the case of other characters.	string ucfirst (string $str)	<?php $str="india is great"; $str=ucfirst($str); echo $str; ?> india is great
ucwords()	The ucwords() function returns string converting first character of each word into uppercase.	string ucwords (string $str)	<?php $str="India is great "; $str=ucwords($str); echo $str; ?> India Is Great
strrev()	The strrev() function returns reversed string.	string strrev (string $string)	<?php $str="INDIA"; $srev=strrev($ str); echo $srev; ?>

trlen()	The strlen() function returns length of the string.	int strlen (string $string)	<?php $str="INDIA"; $l=strlen($str); echo $l; ? >

strpos()	Find the position of the first occurrence of a string inside another string (case- sensitive). If no match is found, it will return FALSE.	Strpos(string, search)	<?php $str="INDIA"; $p=strpos($str, ‖I‖); echo$p; ? > 0
strrpos()	Find the position of the last occurrence of a string inside another string (case- sensitive), if no match is found, it will return FALSE.	Strrpos(string, serach)	<?php $str="INDIA"; $p=strrpos($str ,‖I‖); echo$p; ? >

str_replace()	Replace all occurrences of the search string with the replacement string (case- sensitive)	Str_replace(search, replace,string)	<?php
			echo
			str_replace("world",
			"Dolly", "Hello
			world!"); // outputs
			Hello Dolly!
			?>

strcmp( )	If both strings are equal, strcmp ( ) return 0	Strcmp (String1, String2)	If(strcmp(–India‖,‖indi a‖)== 0) Echo –Both string are equal‖; Else Echo–Both string arenotequal‖;
strstr( )	Find the first occurrence of a string inside another string (case-sensitive)	Strstr(sourceString, subString)	
substr( )	Returns the substring from string that begins at start and is of length characters long.	Substr(string, start, length)	

Exercises

SET A

1. Write a PHP script to accept the number from user and Write a php function to calculate the factorial of a number (a non-negative integer). The function accepts the number as an argument.
2. Design a HTML form to accept a string. Write a php function to reverse a string. 3. Design

a HTML form to accept a string. Write a PHP function that checks whether a passed string is a palindrome or not?

SET B

1. Design a HTML page to accept a number and write a PHP script to display that number in words e.g. 123 - one two three
2. Design a HTML form to accept a string. Write a PHP script for the following. a) Write a function to count the total number of Vowels from the script. b) Show the occurrences of each Vowel from the script.
3. Write a PHP script for the following: a) Design a form to accept two numbers from the users. b) Give option to choose an arithmetic operation (use Radio Button). c) Display the result on next form. d) Use concept of default parameter.

SET C

1. Design a HTML form to accept email address from the user. Write a PHP function 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 pereg function.)

2. 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.)
3. 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)

Assignment Evaluation

0: Not Done [ ] 1: Incomplete [] 2: Late Complete []

3: Needs Improvement [] 4: Complete [] 5: Well-done []

Signature of Instructor:

Date:

Assignment Number 3 :Arrays

Aim: To study use of arrays

Pre-requisite:

· Knowledge of flow control statements, variable declaration, string · Knowledge of functions

Instructions for Teachers/Instructors:

· Demonstration of one dimensional and multi-dimensional arrays with their functions and syntax using simple programmes

Instructions for Students:

· Students must read theory and syntax for solving programmes before his/her practical slot.
· Solve SET A, B or C assigned by instructor in allocated slots only.

Theory:

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

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

In PHP there are three kinds of arrays :

Indexed array / Numeric array : An array with a numeric index starting with 0.

For example,

```
$cars=array("Volvo","BMW","Toyota");
Initializing an indexed array,
$cars[0]= "Volvo";
$cars[1]= "BMW";
$cars[2]= "Toyota";
```

Associative array : An array which have strings as keys which are used to access the values.

For example,

```
$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43");
Initializing an Associative array,
$age[ 'Peter' ]="35";
$age[ 'Ben' ]="37";
$age[ 'Joe' ]="43";
```

Multidimensional array : Arrays containing one or more arrays.
For example,

```
$cars=array(
 array("Volvo",22,18),
 array("BMW",15,13),
 array("Saab",5,2),
```

array(“Land Rover”,17,15)
);

Functions used with array :

Name	Use	Example
array()	This construct is used to 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 array.	$cities=array(‘Capital of Nation‘=>’Delhi’, ’Capital of state’=>’Mumbai’, ’My city’=>’Nashik’); If (array_key_exists(‘Capital of State’,$cities))
		{
		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=’SYBCA’;
foreach()	This is the most common way to loop over elements of an array.	echo “Student $value \n”; } Output Student A Student B

	PHP executes the body of the loop once for each element of $students, with $value set to the current element.	Student C
		Student D
		For associative array :
		$students=array('Name'=>'a','Roll_No' => 100,'Class'=>'SYBCA');
		foreach($students 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 : SYBCA

array_push() array_pop()	These functions are used to treat an array like a stack .	array_push(a); array_pop(a);
array_shift() array_unshift()	These functions are used to treat an array like a queue.	array_shift(); array_unshift();

Exercises

SET A

1. Write a menu driven program to perform the following operations on an associative array:
   a) Display the elements of an array along with the keys.
   b) Display the size of an array

2. Write a menu driven program the following operation on an associative array a) Reverse the order of each element's key-value pair. [Hint: array_flip()] b) Traverse the element in an array in random order. [Hint: shuffle()]

3. Declare array. Reverse the order of elements, making the first element last and last element first and similarly rearranging other array elements.[Hint : array_reverse()]

SET B

1. Declare a Multidimensional Array. Display specific element from a Multidimensional array. Also delete given element from the Multidimensional array. (After each operation display array content.

2. Write a menu driven program to perform the following stack related operations.
   a) Insert an element in stack.
   b) Delete an element from stack.[Hint: array_push(), array_pop()]
3. Write a menu driven program to perform the following queue related operations a) Insert an element in queue
   b) Delete an element from queue
   c) Display the contents of queue

SET C

1. Write a menu driven program to perform the following operations on associative arrays:
    a) Merge the given arrays.
    b) Find the intersection of two arrays.
    c) Find the union of two arrays.
    d) Find set difference of two arrays.

2. Write a menu driven program to perform the following operations on associative arrays:
    a) Sort the array by values (changing the keys) in ascending, descending order.
    b) Also sort the array by values without changing the keys.
    c) Filter the odd elements from an array.
3. Sort the different arrays at a glance using single function.

Assignment Evaluation

0: Not Done [ ] 1: Incomplete [] 2: Late Complete []

3: Needs Improvement [] 4: Complete [] 5: Well Done []

Signature of Instructor:

Date:

Assignment Number 4: Object oriented Programming

Aim: To study object oriented concepts with their use in programming language. Pre-requisite:

Knowledge of flow control statements, variable declaration, string
Knowledge of arrays, functions

Instructions for Teachers/Instructors:

· Demonstration of object oriented concepts and their use in programmes

Instructions for Students:

· Students must read theory and syntax for solving programmes before his/her practical slot.
· Solve SET A, B or C assigned by instructor in allocated slots only.

Theory:

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 be done with 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 class myclass { $instance1=new myclass(); $instance1->mymethod(); ?>

public
$public Memeber =
“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
$protected member
= “Protected
Member”; Private
$private member=
“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 extended
Class 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.	$class = class_exists(classname); $classes = get_declared_classes(); $methods = get_class_methods(classname); $properties=get_class_vars(classn am e); $superclass = get_parent_class (classname);

	This function is used to find the class's parent class.	
s_object() get_class() method_exists() get_object_vars()	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 object exists . This function returns an array of properties set in an object	$obj= is_obj(var); $classname= get_class(object); $method_exists=method_exists(object ,method); $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

Exercises

SET A

1) Write a PHP program to define Interface shape which has two method as area () and volume (). Define a constant PI. Create a class Cylinder implement this interface and calculate area and Volume.
2) a) Write a PHP script to create a Class shape and its subclass triangle, square and display area of the selected shape.(use the concept of Inheritance)
   Display menu (use radio button)
   a) Triangle
   b) Square
   c) Rectangle
   d) Circle

3) Write class declarations and member function definitions for Teacher (code, name, qualification). Derive teach account(account_no,joining_date) from Teacher and teach_sal(basic_pay, earnings, deduction) from each 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 of all teachers.
4) Write PHP script to demonstrate the concept of introspection for examining object.

SET B

1) Create a class account(accno, cust_name). Derive two classes from account as saving_acc(balance, min_amount) and current_acc(balance, min_amount). a) Display a menu
   b) Saving Account
   c) Current Account
   For each of this display a menu with the following options.
   1. Create account
   2. Deposit
   3. Withdrawal

2)Define a class Employee having private members – id, name, department, salary. Define parameterized constructors. Create a subclass called "Manager" with private member bonus. Create 6 objects of the Manager class and display the details of the manager having the maximum total salary (salary + bonus).

SET C

1) Define an interface for queue operation. Implement this interface in a lass.

2) Create an interface for the classes that handle properties, having methods to setProperty() and getProperty() methods. Create another interface HOME having setHasApartment($bool) and getHasSociety() methods which is boolean. Create class Bunglow with two properties set and get and has Garden property and implement the two interfaces.

Assignment Evaluation

0: Not Done [ ] 1: Incomplete [] 2: Late Complete []

3: Needs Improvement [] 4: Complete [] 5: WellDone []

Signature of Instructor:

Date:

Assignment Number 5 : Accessing Databases (PostgreSQL)

Aim: To study creation of database and its use in php programming.

Pre-requisite:

· Knowledge of flow control statements, variable declaration ,string ,arrays, functions.
· Knowledge of object oriented concepts in programming languages · Knowledge of creation of database using PostgreSQL

Instructions for Teachers/Instructors:

· Demonstration of creation of database in 3NF and simple programme to use it.

Instructions for Students:

· Students must read theory and syntax for solving programmes before his/her practical slot.
· Solve SET A, B or C assigned by instructor in allocated slots only. · Create appropriate database for the question asked. i.e. Create the relations accordingly, so that the relationship is handled properly and the relations are in normalized form (3NF).
· Get database checked from the instructor before you start writing PHP script. · Insert sufficient number of records into the database. Make sure result set of any query will be non-empty.

Theory:

Following example shows various functions that used for PostgreSQL database manipulation while writing PHP scripts. With these functions, scripts written by us will access the database for different purposes such as inserting new data, removing data, and fetching data from database. Or even updating the same.

```
<?php
$dbconn = pg_connect("host=localhost dbname=demo user=sybca1
password=sybca1")  or die('Could not connect: ' . pg_last_error());
$query = 'SELECT * FROM employee';
$result = pg_query($query) or die('Query failed: ' .
pg_last_error()); echo "<table>\n";
while ($temp = pg_fetch_array($result, null, PGSQL_ASSOC)) {
 echo "\t<tr>\n";
 foreach ($temp as $col_value) {
 echo "\t\t<td>$col_value</td>\n";
 }
 echo "\t</tr>\n";
```

Reference: -

https://www.php.net/manual/ }

```
echo "</table>\n";
pg_free_result($result);
pg_close($dbconn);
?>
```

Function name	Description
pg_connect ()	pg_connect (string $connection_string [, int $connect_type]): resource Purpose: opens a connection to a PostgreSQL database specified by the connection_string
pg_close ()	pg_close ([resource$connection]): bool Purpose: closes the non-persistent connection to a PostgreSQL database associatedwith the given connection resource.
pg_prepare ()	pg_prepare ([resource $connection], string $stmtname, string $query): resource Purpose: Submits a request to create a prepared statement with the given parameters, and waits for completion
pg_execute ()	pg_execute ([resource $connection], string $stmtname, array $params): resource Purpose: Sends a request to execute a prepared statement with given parameters, and waits for the result
pg_query ()	pg_query ([resource $connection], string $query): resource Purpose: pg_query () executes the query on the specified database connection. pg_query_params () should be preferred in most cases. If an error occurs, and FALSE is returned, details of the error can be retrieved using the pg_last_error () function if the connection is valid.
pg_fetch_assoc ()	pg_fetch_assoc (resource $result [, int $row]): array Purpose: returns an associative array that corresponds to the fetched row (records).
pg_fetch_row ()	pg_fetch_row (resource $result [, int $row]): array Purpose: fetches one row of data from the result associated with the specified result resource.

Apart from these there are various other functions also supported/provided.

· pg_field_is_null() — Test if a field is SQL NULL
· pg_field_name() — Returns the name of a field
· pg_port()— Return the port number associated with the connection ·
. pg_version() — Returns an array with client, protocol and server version (when available)

Exercises

SET A

1. Property (pno, description, area)
    a. Owner (oname, address, phone)
    b. An owner can have one or more properties, but a property belongs to exactly one owner.
    c. Accept owner name from the user. Write a PHP script which will display all properties which are own by that owner.

2. Sales_order (sonumber, s_order_date)
    a. Client (clientno, name, address)
    b. A client can give one or more sales_orders, but a sales_order belongs to exactly one client.
    c. Accept sales order date from the user. Write a PHP script which will display all orders which are placed before that date.
3. Emp (eno, ename, edesignation, esalary)
   Dept (dno, dname, dlocation)
   There exists a one-to-many relationship between Emp and Dept.
   Accept department name from the user. Write a PHP script which will display count of the employees working in that department.

SET B

1. Project (pno, pname, pduration, pbudget)
    a. You have already created relation Emp in SET A-3. There exists a many to-many relationship between Emp and Project, with descriptive attribute no_of_hrs_worked.
    b. Accept project name from the user. Write a PHP script which will display all employees working on that project along with number of hours they worked on it.

2. Refer database created in SET A- 3.
    a. Accept department name form the user. Write a PHP script which will display maximum salary, minimum salary, and sum of salary for a given department. Format of the result shown should be like –

Department name: ……		
Maximum Salary	Minimum Salary	Sum of the Salary

3. Doctor (doc_no, doc_name, experience, city, area)
   Hospital (hosp_no, hosp_name, hosp_city)
   Doctor and Hospital are related with many-many relationship with descriptive attribute type_of_appointment. (It can be either visiting or working)

Write a PHP script which accepts experience value from the user and update city to Mumbai for all doctors whose experience is less than entered value.

SET C

An insurance agent sells policies to clients. Each policy is of a particular type like vehicle insurance, life insurance, accident insurance etc, and there can be many policies of a particular type. Each policy will have many monthly premiums, and each premium is associated to only one policy. Assume appropriate attributes for agents, policy, premiums, policy-types.
The following constraints have to be defined on the relations.

a. The policy types can be only accident, life, vehicle.
b. The agents can be only from Pune, Mumbai, Chennai.
c. The policy amount should be greater than 20000.
d. The policy-sale-date should be greater than the policy-intro-date. Using database created for the above case-study, solve the following. *Use appropriate error message if input entered by the user is incorrect.

* Write a PHP Script for –

1. Accept minimum years of experience from the user. Display agent details such as name, city he belongs to and experience in tabular format who are having at-least that much experience.
2. Accept agent name from the user. Display all policy holders those who have bought policies from that agent.
3. Accept details of new agent and add that record into database.

OR

1. Accept policy type from the user using radio button. Display all policy holders along with details such as name, birthdate in the tabular format who have bought policies of that type.
2. Accept policy amount from the user. Display count of policies with at-least that amount.
3. Accept details of policy number and remove that record from the database.

OR

1. Display city names which has maximum number of agents.
2. Accept policy number from the user. Display all details of premiums paid so far, for that policy. Output should be in tabular format and it should contain policy holder name.
3. Display the latest policy introduced.

Assignment Evaluation

0: Not Done [ ] 1: Incomplete [] 2: Late Complete []

3: Needs Improvement [] 4: Complete [] 5: Well-done []

Signature of Instructor:

Date: