ACTIVITY ANSWER SHEET

Name	Rije J.Ladao
Section:	3R1

Instructions:

1. Push your output on your GITHUB repository.
2. Use the answer sheet provided save it as PDF file then push it to your GitHub.
3. Answer the ff. problems write it on the answer sheet.
4. Late submissions will no longer be accepted.
5. Caught copying outputs of others will be given sanctions.
6. Failure to follow these instructions will be given sanctions.

Activity 1: Control Structures

1. Write down the syntax in PHP for the ff.

1. if	`if (condition) {`
	`    code to be executed if condition is true;`
	`}`

2. if…else	`if (condition) {`
	`    code to be executed if condition is true;`
	`} else {`
	`    code to be executed if condition is false;`
	`}`

3. if…else if…else	`if (condition) {`
	`    code to be executed if this condition is true;`
	`} elseif (condition) {`
	`    code to be executed if first condition is false and this condition is true;`
	`} else {`
	`    code to be executed if all conditions are false;`
	`}`

4. switch…case	`switch (n) {`
	`    case label1:`
	`        code to be executed if n=label1;`
	`        break;`
	`    case label2:`
	`        code to be executed if n=label2;`
	`        break;`
	`    case label3:`
	`        code to be executed if n=label3;`
	`        break;`
	`    ...`
	`    default:`
	`        code to be executed if n is different from all labels;`
	`}`

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

6. do while loop	`do {`
	`    code to be executed;`
	`} while (condition is true);`

7. while loop	`while (condition is true) {`
	`    code to be executed;`
	`}`

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

9. break statement	jump statement;
	break;

10. continue statement	while (expression 1)
	{
	if (expression 2)
	{
	continue ;
	}
	// Operation Statements
	}

11. try…catch	`<?php`
	`try {`
	`    //code goes here that could potentially throw an exception`
	`}`
	`catch (Exception $e) {`
	`    //exception handling code goes here`
	`}`
	`?>`

2. Solve the ff. problem using PHP.

a. Write a program that checks if value is a number (integer).

Sample input: ‘1’	Sample input: 1
Expected output: Not a number	Expected output: A number

Answer:

```
<form action="index.php" method="post" >
      Value: <input type="text" name="value" onClick='clearform();' />
   </form>
<?php
     if (isset($_POST['value'])) {
        $Value = ($_POST['value']);
     }
         if (filter_var($Value, FILTER_VALIDATE_INT)===0||!filter_var($Value,
FILTER_VALIDATE_INT)===FALSE) {
        echo "A number<br>";
      }else{
            echo "Not a number";
        }
?>
```

b. Write a program that checks if a value is positive or negative and odd or even.

Sample input: 0	Sample input: -1
Expected output: Positive & Even	Expected output: Negative and Odd

Answer:

```
  <form action="index.php" method="post" >
     Number: <input type="text" name="Number" onClick='clearform();' />
   </form>
<?php
if (isset($_POST['Number'])) {
   $Number = ($_POST['Number']);}
if (filter_var($Number, FILTER_VALIDATE_INT)===0||!filter_var($Number,
FILTER_VALIDATE_INT)===FALSE) {
    if ($Number%2==0 && $Number>=0) {
      echo "Positive & Even";
   }elseif ($Number%2==1 && $Number>0) {
      echo "Posive & Odd";
   }elseif ($Number%2==0 && $Number<0) {
       echo "Negative & even";
    }elseif ($Numbere%2==-1) {
      echo "Negative & Odd";
   }}
else {
      echo "Not a number please enter a number again";
   }
?>
```

c. Write a program that checks if a value is palindrome.

Sample input: Anna   Sample input: Bogart
Expected output: Palindrome   Expected output: Not a Palindrome

Answer:

```
 <form action="index.php" method="post" >
     word: <input type="text" name="word" onClick='clearform();' />
   </form>
<?php
if (isset($_POST['word'])) {
   $word = ($_POST['word']);}
   if ($word==(strrev($word))){
      echo "Palindrome";
   }else {
      echo "Not a Palindrome";
   }
?>
```

d. Write a program to calculate and print the factorial of a number using a for loop.

Sample input: 4
Expected output: 24

Answer:

```
<form action="index.php" method="post" >
      number: <input type="text" name="number" onClick='clearform();' />
   </form>
<?php
if (isset($_POST['number'])) {
   $number = ($_POST['number']);}
   $factorial=1;
   for ($i=1; $i <= $number; $i++) {
      $factorial=$factorial*$i;
   }
   echo ($factorial);
?>
```

e. Write a PHP program to generate and display the first n lines of a Floyd triangle.

Sample input: 3
Sample output:

```
1
2 3
4 5 6
```

Answer:

```
<?php
echo “<pre> ”;
$num =1;
for ($i =1; $i <=8; $i++){
		for ($b =1; $b <=$i ; $b++){
				echo $num . “&nbsp;”;
				$num++;
				If ($b == $i ){
						echo ”&nbsp;”;
						echo “< br/ >”;
				}
		}
}
echo “< /pre>”;
?>
```

Activity 2: PHP Built-in Functions

Write down the functionalities of the ff. built-in functions in PHP. Provide 5 example functions for each built-in PHP function.

Array	The array functions allow you to access and manipulate arrays.Simple and multi-dimensional arrays are supported. Example: array() array_change_key_case() ,array_chunk(

	) ,array_column() ,array_combine()
Calendar	The calendar extension contains functions that simplifies converting between different calendar formats. It is based on the Julian Day Count, which is a count of days starting from January 1st, 4713 B.C. Example: cal_days_in_month() ,cal_from_jd() cal_info() ,cal_to_jd() easter_date() Returns the Unix timestamp for midnight on Easter of a specified year
Date	PHP date function is an in-built function that simplify working with date data types. The PHP date function is used to format a date or time into a human readable format. It can be used to display the date of article was published. record the last updated a data in a database. Example: checkdate() Validates a Gregorian date date_add() Adds days, months, years, hours, minutes, and seconds to a date date_create_from_format() Returns a new DateTime object formatted according to a specified format date_create() Returns a new DateTime object date_date_set() Sets a new date
Directory	The directory functions allow you to retrieve information about directories and their contents. Example: chdir() Changes the current directory chroot() Changes the root directory closedir() Closes a directory handle dir() Returns an instance of the Directory class getcwd() Returns the current working directory
Error	The error functions are used to deal with error handling and logging.The error functions allow us to define own error handling rules, and modify the way the errors can be logged.The logging functions allow us to send messages directly to other machines, emails, or system logs. The error reporting functions allow us to customize what level and kind of error

	feedback is given. Example: debug_backtrace() Generates a backtrace debug_print_backtrace() Prints a backtrace error_clear_last() Clears the last error error_get_last() Returns the last error that occurred error_log() Sends an error message to a log, to a file, or to a mail account
File System	The filesystem functions allow you to access and manipulate the filesystem. Example: basename() Returns the filename component of a path chgrp() Changes the file group chmod() Changes the file mode chown() Changes the file owner clearstatcache() Clears the file status cache
Filter	PHP filters are used to validate and sanitize external input. The PHP filter extension has many of the functions needed for checking user input, and is designed to make data validation easier and quicker. Example: filter_has_var() Checks whether a variable of a specified input type exist filter_id() Returns the filter ID of a specified filter name filter_input() Gets an external variable (e.g. from form input) and optionally filters it filter_input_array() Gets external variables (e.g. from form input) and optionally filters them filter_list() Returns a list of all supported filter names
FTP	PHP filters are used to validate and sanitize external input. The PHP filter extension has many of the functions needed for checking user input, and is designed to make data validation easier and quicker.

	Example: ftp_alloc() Allocates space for a file to be uploaded to the FTP server ftp_cdup() Changes to the parent directory on the FTP server ftp_chdir() Changes the current directory on the FTP server ftp_chmod() Sets permissions on a file via FTP ftp_close() Closes an FTP connection
Libxml	The libxml functions and constants are used together with SimpleXML, XSLT and DOM functions. Example: libxml_clear_errors() Clears the libxml error buffer libxml_disable_entity_loader() Enables the ability to load external entities libxml_get_errors() Gets the errors from the the libxml error buffer libxml_get_last_error() Gets the last error from the the libxml error buffer libxml_set_external_entity_loader() Changes the default external entity loader
Mail	The mail() function is used to send a mail. Example: ezmlm_hash() Calculates the hash value needed by EZMLM mail() Allows you to send emails directly from a script
Math	The math functions can handle values within the range of integer and float types. Example: abs() Returns the absolute (positive) value of a number acos() Returns the arc cosine of a number acosh() Returns the inverse hyperbolic cosine of a number asin() Returns the arc sine of a number atan() Returns the arc tangent of a number in radians
Misc	The misc. functions were only placed here because none of the other categories seemed to fit. Example:

	connection_aborted() Checks whether the client has disconnected connection_status() Returns the current connection status connection_timeout() Deprecated from PHP 4.0.5. Checks whether the script has timed out constant() Returns the value of a constant define() Defines a constant
MySQLi	The MySQLi functions allows you to access MySQL database servers. Example: affected_rows() Returns the number of affected rows in the previous MySQL operation autocommit() Turns on or off auto-committing database modifications begin_transaction() Starts a transaction change_user() Changes the user of the specified database connection character_set_name() Returns the default character set for the database connection
Network	The Network functions contains various network function and let you manipulate information sent to the browser by the Web server, before any other output has been sent. Example: checkdnsrr() Checks DNS records for *type* corresponding to *host* closelog() Closes the connection of system logger define_syslog_variables() Deprecated and removed in PHP 5.4. Initializes the variables used in syslog functions dns_check_record() Alias of checkdnsrr() dns_get_mx() Alias of getmxrr()
SimpleXML	SimpleXML is an extension that allows us to easily manipulate and get XML data.SimpleXML provides an easy way of getting an element's name, attributes and textual content if you know the XML document's structure or layout.SimpleXML turns an XML document into a data structure you can iterate through like a collection of arrays and objects. Example: __construct() Creates a new SimpleXMLElement object __toString() Returns the string content of an element addAttribute() Appends an attribute to the SimpleXML element addChild() Appends a child element the

	SimpleXML element asXML() Returns a well-formed XML string (XML version 1.0) from a SimpleXML object
Stream	The Stream functions .... Streams are the way of generalizing file, network, data compression, and other operations which share a common set of functions and uses. In its simplest definition, a stream is a resource object which exhibits streamable behavior. That is, it can be read from or written to in a linear fashion, and may be able to fseek() to an arbitrary location within the stream. A wrapper is additional code which tells the stream how to handle specific protocols/encodings. Example: set_socket_blocking(), stream_bucket_prepend(), stream_context_create(), stream_context_get_default(), stream_context_get_options()
String	A string is a sequence of characters, like "Hello world!". Example: addcslashes() Returns a string with backslashes in front of the specified characters addslashes() ,bin2hex() , chop(),chr(),
XML Parser	The XML functions lets you parse, but not validate, XML documents. XML is a data format for standardized structured document exchange. Example: utf8_decode() Decodes an UTF-8 string to ISO-8859-1 utf8_encode() Encodes an ISO-8859-1 string to UTF-8 xml_error_string() Returns an error string from the XML parser xml_get_current_byte_index() Returns the current byte index from the XML parser xml_get_current_column_number() Returns the current column number from the XML parser

Zip	The Zip files functions allows you to read ZIP files. Example: zip_close() Closes a ZIP file archive zip_entry_close() Closes a ZIP directory entry zip_entry_compressedsize() Returns the compressed file size of a ZIP directory entry zip_entry_compressionmethod() Returns the compression method of a ZIP directory entry zip_entry_filesize() Returns the actual file size of a ZIP directory entry
Timezones	The date_default_timezone_set() function is an inbuilt function in PHP which is used to set the default timezone used by all date/time example:functions in a script. This function returns False if the timezone is not valid, or True otherwise

Activity 3: Regular Expression

1. Define Regular Expression (RegEx) and provide example programming scenario where you can use (RegEx). Provide example syntax in PHP.

-Regular expressions commonly known as a regex (regexes) are a sequence of characters describing a special search pattern in the form of text string. They are basically used in programming world algorithms for matching some loosely defined patterns to achieve some relevant tasks. Some times regexes are understood as a mini programming language with a pattern notation which allows the users to parse text strings. The exact sequence of characters are unpredictable beforehand, so the regex helps in fetching the required strings based on a pattern definition.

Programming scenario:

- Regular expressions help in validation of text strings which are of programmer's interest.
- Regexes are very useful for creation of HTML template system recognizing tags.
- Regexes are mostly used for browser detection, spam filteration, checking password strength and form validations.

Example syntax in php:

```
<?php

// Declare a regular expression
$regex = "/<b>(.*)<\/b>/U";

// Declare a string
$inputString = "Name: <b>John</b> Position: <b>Developer</b>";

// Use preg_match_all() function to perform
// a global regular expression match
preg_match_all($regex, $inputString, $output);

echo $output[0][0]." <br> ".$output[0][1]."\n";

?>
```

2. Solve the ff. problem using Regular Expressions.

a. Write a PHP script that checks if a string contains another string

Sample String: 'The quick brown fox'
Test input: 'Fox'
Expected output: Fox is found the string

Code:

```
 <form action="index.php" method="post" >
      test: <input type="text" name="test" onClick='clearform();' />
   </form>
<?php
if (isset($_POST['test'])) {
   $test = ($_POST['test']);}
   $SampleStr="The quick brown fox";
   if (strpos($SampleStr, $test) == true) {
      echo "$test is found the string";
   }else {  echo "$test not found"; } ?>
```

b. Write a PHP script that removes the last word from a string.
Sample String: 'The quick brown fox'
Expected output: ‘The quick brown'

Code:

```
<?php
   $SampleStr="The quick brown fox";
   $word= explode(" ",$SampleStr);
   array_splice($word,-1);
   echo implode(" ",$word);
?>
```

c. Write a PHP script to remove nonnumeric characters except comma and dot.
Sample String: '/$123,34.00A#'
Expected output: 123,34.00

Code:

```
<?php
   $SampleStr="'/$123,34.00A#";
   echo preg_replace("/[^0-9,.]/","",$SampleStr);
?>
```

d. Write a PHP script to extract text (within parenthesis) from a string.
Sample String: ‘The quick brown [fox].’
Expected output: Fox

Code:

```
<?php
   $SampleStr='The quick brown (fox)';
   $openparenthesis= strpos($SampleStr,"(" );
   $closeparenthesis= strpos($SampleStr,")" );
   echo substr($SampleStr,$openparenthesis+1, $closeparenthesis-$openparenthesis-
1);

?>
```

e. Write a PHP script to remove all characters from a string except a-z A-Z 0-9 or " ".
Sample String: ‘abcde$ddfd @abcd)der]’
Expected output: abcdeddfd abcd der

Code:

```
<?php
   $SampleStr='abcde$ddfd @abcd )der]';
   echo preg_replace("/[^0-9 ^a-z^A-Z]/","",$SampleStr);
?>
```

Activity 4: Error Handling

1. List down the different PHP errors. Provide example code on how to handle these errors.

There are six types of error present in PHP

- E_WARNING :- Non-fatal run-time errors. Execution of the script is not terminated
- E_NOTICE :- Run-time notices. The script found something that might be an error, but could also happen when running a script normally
- E_USER_ERROR :- user-generated error. This is like an E_ERROR set by the programmer using the PHP function trigger_error()Fatal

- E_USER_WARNING :- user-generated warning. This is like an E_WARNING set by the programmer using the PHP function trigger_error()Non-fatal
- E_USER_NOTICE :- User-generated notice. This is like an E_NOTICE set by the programmer using the PHP function trigger_error()
- E_ALL :- All errors and warnings (E_STRICT became a part of E_ALL in PHP 5.4)

There are three basic methods when it comes to PHP error handling:

- A basic `die()` statement.
- Defining your own error messages and alerts (making PHP show errors).
- Reporting errors.

Using die() Function

```
<?php
$file=fopen("mytestfile.txt","r");
?>
```

If the file does not exist you might get an error like this:

```
Warning: fopen(mytestfile.txt) [function.fopen]: failed to open stream:
No such file or directory in C:\webfolder\test.php on line 2
```

To prevent the user from getting an error message like the one above, we test whether the file exist before we try to access it:

```
<?php
if(file_exists("mytestfile.txt")) {
  $file = fopen("mytestfile.txt", "r");
} else {
  die("Error: The file does not exist.");} ?>
```