

ACTIVITY ANSWER SHEET

Name	Aceron, Prisy P.
Section:	BS IT-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	<pre> if (condition) { code to be executed if condition is true; } </pre>
2. if...else	<pre> if (condition) { code to be executed if condition is true; } else { code to be executed if condition is false; } </pre>
3. if...else if...else	<pre> if (condition) { code to be executed if this condition is true; } elseif (condition) { code to be executed if this condition is true; } else { code to be executed if all conditions are false; } </pre>
4. switch...case	<pre> switch (n) { case label1: code to be executed if n=label1; break; case label2: code to be executed if n=label2; break; default : code to be executed if n is different from all labels; } </pre>
5. for loop	for (init counter; test

	counter; increment counter) { code to be executed; }
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	break;
10. continue statement	continue;
11. try...catch	try { //run your code here } catch (Exception \$e) { echo \$e->getMessage(); }

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

```
<?php
$p='7';
if (is_integer($p)){
    echo "A number";
} 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

```
<?php
$p='7';
if ($p < 0){
    if ($p%2 ==0){
        echo "Negative and Even";
    }else {
        echo "Negative and Odd";
    }
} else {
    if ($p%2 == 0){
        echo "Positive and Even";
    }else
        echo "Positive and Odd";
}
```

```
?>
```

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

```
<?php
palindrome("dad");

function palindrome($drome) {
    $drome_len = strlen($drome) - 1;
    $result = "";

    for ($y = $drome_len; $y >= 0; $y--) {
        $result .= $drome[$y];
    }
    if ($result == $drome){
        echo "Palindrome";
    }else {
        echo "Not Palindrome";
    }
}

?>
```

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

Sample input: 4

Expected output: 24

```
<?php
$f=5;
$c=1;
for ($a=1; $a<=$f;$a++)
{
    $c=$c*$a;
}
echo "Factorial =".$c;

?>
```

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
```

```
<?php
$f=5;
$l=1;
$o=0;
$y=0;
for ($y= $f; $y> 0; $y--) {
    for ($o=$y; $o<$f; $o++){
        printf("%4o", $l);
```

```

        $l++;

    }

    echo nl2br("\n");

}

?>

```

Activity 2: PHP Built-in Functions

Write down the functionalities of the ff. built-in functions in PHP.

Array	<p>These functions allow you to interact with and manipulate arrays in various ways. Arrays are essential for storing, managing, and operating on sets of variables.</p> <p>next() -Advance the internal array pointer of an array.</p> <p>range() -Creates an array containing a range of elements.</p> <p>list() -Assigns variables as if they were an array.</p> <p>asort() -Sorts an associative array in ascending order, according to the value.</p> <p>end() -Sets the internal pointer of an array to its last element.</p>
Calendar	<p>The calendar extension presents a series of functions to simplify converting between different calendar formats.</p> <p>jddayofweek() -Returns the day of the week.</p> <p>jdmonthname() -Returns a month name.</p>

	<p>cal_info() -Returns information about a specified calendar.</p> <p>cal_days_in_month() -Returns the number of days in a month for a specified year and calendar.</p> <p>cal_from_jd() -Converts a Julian Day Count into a date of a specified calendar.</p>
Date	<p>These functions allow you to get the date and time from the server where your PHP scripts are running. You can use these functions to format the date and time in many different ways.</p> <p>date_create() -Returns a new DateTime object.</p> <p>date_date_set() -Sets a new date.</p> <p>date_default_timezone_set() -Sets the default timezone used by all date/time functions.</p> <p>date_diff() -Returns the difference between two dates.</p> <p>date_format() -Returns a date formatted according to a specified format.</p>
Directory	<p>These functions are provided to manipulate any directory.</p> <p>dir() -Returns an instance of the Directory class.</p> <p>opendir() -Opens a directory handle.</p> <p>readdir() -Returns an entry from a directory handle.</p> <p>rewinddir() -Resets a directory handle.</p> <p>chdir() -Changes the current directory.</p>
Error	<p>These are functions dealing with error handling and logging. They allow you to define your own error handling rules, as well as modify the way the errors can be logged. This allows you to change and enhance error reporting to suit your needs.</p> <p>error_reporting() -Specifies which errors are reported</p> <p>restore_error_handler() -Restores the previous error handler</p> <p>restore_exception_handler() -Restores the previous exception handler</p> <p>set_error_handler() -Sets a user-defined error handler function</p> <p>set_exception_handler() -Sets a user-defined exception handler function.</p>
File System	<p>The file system functions are used to access and</p>

	<p>manipulate the file system PHP provides you all the possible functions you may need to manipulate a file.</p> <p>filegroup() -Returns the group ID of a file fileinode() -Returns the inode number of a file filemtime() -Returns the last modification time of a file fileowner() -Returns the user ID (owner) of a file fileperms() -Returns the file's permissions.</p>
Filter	<p>This PHP filters is used to validate and filter data coming from insecure sources, like user input.</p> <p>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 filter_var() -Filters a variable with a specified filter.</p>
FTP	<p>The FTP functions give client access to file servers through the File Transfer Protocol (FTP). The FTP functions are used to open, login and close connections, as well as upload, download, rename, delete, and get information on files from file servers.</p> <p>ftp_chmod() -Sets permissions on a file via FTP ftp_close() -Closes an FTP connection ftp_connect() -Opens an FTP connection ftp_delete() -Deletes a file on the FTP server ftp_exec() -Executes a command on the FTP server.</p>
Libxml	<p>The libxml functions and constants are used together with SimpleXML, XSLT and DOM functions.</p> <p>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 libxml_set_streams_context() -Sets the streams</p>

	<p>context for the next libxml document load or write</p> <p>libxml_use_internal_errors() -Disables the standard libxml errors and enables user error handling</p>
Mail	<p>The mail() function allows you to send emails directly from a script.</p> <p>ezmlm_hash() -Calculates the hash value needed by EZMLM</p> <p>mail() -Allows you to send emails directly from a script.</p>
Math	<p>The math functions can handle values within the range of integer and float types.</p> <p>is_infinite() -Checks whether a value is infinite or not</p> <p>is_nan() -Checks whether a value is 'not-a-number'</p> <p>lcg_value() -Returns a pseudo random number in a range between 0 and 1</p> <p>log() -Returns the natural logarithm of a number</p> <p>log10() -Returns the base-10 logarithm of a number.</p>
Misc	<p>The misc. functions were only placed here because none of the other categories seemed to fit.</p> <p>sys_getloadavg() -Returns the system load average</p> <p>time_nanosleep() -Delays code execution for a number of seconds and nanoseconds</p> <p>time_sleep_until() -Makes a script sleep until the specified time</p> <p>uniqid() -Generates a unique ID</p> <p>unpack() -Unpacks data from a binary string</p>
MySQLi	<p>The MySQLi functions allows you to access MySQL database servers.</p> <p>get_connection_stats() -Returns statistics about the client connection</p> <p>get_host_info() -Returns the MySQL server hostname and the connection type</p> <p>get_proto_info() -Returns the MySQL protocol version</p> <p>get_server_info() -Returns the MySQL server</p>

	<p>version</p> <p>get_server_version() -Returns the MySQL server version as an integer</p>
Network	<p>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.</p> <p>setrawcookie() -Defines a cookie (without URL encoding) to be sent along with the rest of the HTTP headers</p> <p>socket_get_status() -Alias of stream_get_meta_data()</p> <p>socket_set_blocking() -Alias of stream_set_blocking()</p> <p>socket_set_timeout() -Alias of stream_set_timeout()</p> <p>syslog() -Generates a system log message</p>
SimpleXML	<p>SimpleXML is an extension that allows us to easily manipulate and get XML data.</p> <p>saveXML() -Alias of asXML()</p> <p>simplexml_import_dom() -Returns a SimpleXMLElement object from a DOM node</p> <p>simplexml_load_file() -Converts an XML document to an object</p> <p>simplexml_load_string() -Converts an XML string to an object</p> <p>xpath() -Runs an XPath query on XML data</p>
Stream	<p>Streams are the way of generalizing file, network, data compression, and other operations which share a common set of functions and uses.</p> <p>stream_register_wrapper() -Alias of stream_wrapper_register()</p> <p>stream_copy_to_stream() -Copies data from one stream to another</p> <p>stream_bucket_prepend()</p> <p>stream_context_create()</p> <p>stream_filter_append() -Appends a filter to a stream</p>
String	<p>The PHP string functions are part of the PHP core. No installation is required to use these functions.</p>

	<p>crypt() -One-way string hashing</p> <p>echo() -Outputs one or more strings</p> <p>explode() -Breaks a string into an array</p> <p>fprintf() -Writes a formatted string to a specified output stream</p> <p>count_chars() -Returns information about characters used in a string</p>
XML Parser	<p>The XML functions lets you parse, but not validate, XML documents.</p> <p>xml_parse() -Parses an XML document</p> <p>xml_parse_into_struct() -Parses XML data into an array</p> <p>xml_parser_create_ns() -Creates an XML parser with namespace support</p> <p>xml_parser_create() -Creates an XML parser</p> <p>xml_parser_free() -Frees an XML parser</p>
Zip	<p>The Zip files functions allows you to read ZIP files.</p> <p>zip_entry_name() -Returns the name of a ZIP directory entry</p> <p>zip_entry_open() -Opens a directory entry in a ZIP file for reading</p> <p>zip_entry_read() -Reads from an open directory entry in the ZIP file</p> <p>zip_open() -Opens a ZIP file archive</p> <p>zip_read() -Reads the next file in a open ZIP file archive</p>
Timezones	<p>date_default_timezone_get()</p> <p>Returns the default timezone used by all functions</p> <p>date_default_timezone_set()</p> <p>Sets the default timezone</p> <p>Strtotime()</p> <p>Gmtime()</p>

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 are nothing more than a sequence or pattern of characters itself. They provide the foundation for pattern-matching functionality. Using regular expression you can search a particular string inside a another string, you can replace one string by another string and you can split a string into many chunks.

-When checking that an email address entered into a form.

-<?php

function_name('/pattern/',subject);

?>

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

```
<?php
$string = "The quick brown fox";
$testing = "/Fox/i";
if (preg_match($testing, $string))
{
    echo "Fox is found in the string";
}
else
{
    echo "Fox is not found in the string";
}
?>
```

- b. Write a PHP script that removes the last word from a string.

Sample String: 'The quick brown fox'

Expected output: 'The quick brown'

```
<?php
$string = "The quick brown fox";
echo preg_replace('/\W\w+\s*(\W*)$/','',$1', $string)."\n";
?>
```

- c. Write a PHP script to remove nonnumeric characters except comma and dot.

Sample String: '\$123,34.00A#'

Expected output: 123,34.00

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

- d. Write a PHP script to extract text (within parenthesis) from a string.

Sample String: 'The quick brown [fox].'

Expected output: Fox

```
<?php
$str = 'The quick brown [fox].';
preg_match('#\[.*?\]#', $str, $match);
print $match[1]."\n";
?>
```

- 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

```
<?php
$alphabet = 'abcde$ddfd @abcd )der]';
$run = preg_replace("/[^A-Za-z0-9 ]/", "", $alphabet);
echo 'Output : '.$run."\n";
?>
```

Activity 4: Error Handling

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

***Parse Errors**

```
try{
eval("echo 'toto' echo 'tata'");
```

```
}catch(ParseError $p){
```

```
    echo $p->getMessage();
}
```

***Fatal Errors**

```
set_error_handler('myErrorHandler');
```

```
register_shutdown_function('fatalErrorShutdownHandler');
```

```
function myErrorHandler($code, $message, $file, $line) {
```

```
    ...
}
```

```
function fatalErrorShutdownHandler()
```

```
{
```

```
    $last_error = error_get_last();
```

```
    if ($last_error['type'] === E_ERROR) {
```

```
        // fatal error
```

```
        myErrorHandler(E_ERROR, $last_error['message'], $last_error['file'], $last_error['line']);
```

```
    }
```

```
}
```

***Warning Errors**

```
set_error_handler("warning_handler", E_WARNING);
```

```
dns_get_record(...)
```

```
restore_error_handler();
```

```
function warning_handler($errno, $errstr) {
```

```
    // do something
```

***Notice Errors**

```
<?php
```

```
// Turn off all error reporting
```

```
error_reporting(0);
```

```
// Report simple running errors
```

```
error_reporting(E_ERROR | E_WARNING | E_PARSE);
```

```
// Reporting E_NOTICE can be good too (to report uninitialized
```

```
// variables or catch variable name misspellings ...)
```

```
error_reporting(E_ERROR | E_WARNING | E_PARSE | E_NOTICE);
```

```
// Report all errors except E_NOTICE
```

```
error_reporting(E_ALL & ~E_NOTICE);
```

```
// Report all PHP errors (see changelog)
```

```
error_reporting(E_ALL);
```

```
// Report all PHP errors
```

```
error_reporting(-1);
```

```
// Same as error_reporting(E_ALL);
```

```
ini_set('error_reporting', E_ALL);
```

```
?>
```