

ACTIVITY ANSWER SHEET

Name	Cindy P. Jalop
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	<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 first condition is false and 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; case label3: code to be executed if n=label3; break; ... default: code to be executed if n is different from all labels; }</pre>
5. for loop	<pre>for (init counter; test counter; increment counter) { code to be executed for each iteration; }</pre>
6. do while loop	<pre>do { code to be executed; } while (condition is true);</pre>
7. while loop	<pre>while (condition is true) { code to be executed; }</pre>
8. foreach loop	<pre>foreach (\$array as \$value) { code to be executed; }</pre>

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

```
<?php
$int = 1;

if (filter_var($int, FILTER_VALIDATE_INT) === 1 || filter_var($int, FILTER_VALIDATE_INT)) {
    echo("its 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

```
<form method="post">
Enter Your number<input type="text" name="n"/><hr/>
<input type="submit" value="check number"/>
</form>
<?php
$num=$_POST['n'];

if($num% 2==0 && $num>=0){
echo "positive and even number";
}elseif($num%2==0 && $num<0){
    echo "negative and even number";
}elseif($num%2==1 && $num<0){
```

```

        echo "negative and odd number";
}elseif ($num%2==1 && $num>=0){
    echo "positive and odd number";
}else{
    echo "not a number";
}
?>

```

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

$str = 'maam';
$strLen = strlen($str)-1;
$revStr = '';
for($i=$strLen; $i>=0; $i--){
    $revStr.=$str[$i];
}
if($revStr == $str)
    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
$n = 6;
$x = 1;
for($i=1;$i<=$n-1;$i++)
{
    $x*=( $i+1);
}
echo "The factorial of  $n = $x"."\\n";
?>

```

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
echo "<pre>";
$num = 1;
for ($a = 1; $a <= 8; $a++) {
    for ($b = 1; $b <= $a; $b++) {
        echo $num . "&nbsp;";
        $num++;
        if ($b == $a) {
            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	<p>The array functions allow you to access and manipulate arrays.Simple and multi-dimensional arrays are supported.</p> <p>Example:</p> <p>array() Creates an array</p> <p>array_change_key_case() Changes all keys in an array to lowercase or uppercase</p> <p>array_chunk() Splits an array into chunks of arrays</p> <p>array_column() Returns the values from a single column in the input array</p> <p>array_combine() Creates an array by using the elements from one "keys" array and one "values" array</p>
Calendar	<p>The calendar extension contains functions that simplifies converting between different calendar formats.</p> <p>It is based on the Julian Day Count, which is a count of days starting from January 1st, 4713 B.C.</p> <p>Example:</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> <p>cal_info() Returns information about a specified calendar</p> <p>cal_to_jd() Converts a date in a specified calendar to Julian Day Count</p> <p>easter_date() Returns the Unix timestamp for midnight on Easter of a specified year</p>
Date	<p>PHP date function is an in-built function that simplify working with date data types. The PHP date function is used to</p>

	<p>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.</p> <p>Example:</p> <p>checkdate() Validates a Gregorian date</p> <p>date_add() Adds days, months, years, hours, minutes, and seconds to a date</p> <p>date_create_from_format() Returns a new DateTime object formatted according to a specified format</p> <p>date_create() Returns a new DateTime object</p> <p>date_date_set() Sets a new date</p>
Directory	<p>The directory functions allow you to retrieve information about directories and their contents.</p> <p>Example:</p> <p>chdir() Changes the current directory</p> <p>chroot() Changes the root directory</p> <p>closedir() Closes a directory handle</p> <p>dir() Returns an instance of the Directory class</p> <p>getcwd() Returns the current working directory</p>
Error	<p>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.</p> <p>The error reporting functions allow us to customize what level and kind of error feedback is given.</p> <p>Example:</p> <p>debug_backtrace() Generates a backtrace</p> <p>debug_print_backtrace() Prints a backtrace</p> <p>error_clear_last() Clears the last error</p> <p>error_get_last() Returns the last error that occurred</p> <p>error_log() Sends an error message to a log, to a file, or to a mail account</p>
File System	<p>The filesystem functions allow you to access and manipulate the filesystem.</p> <p>Example:</p> <p>basename() Returns the filename component of a path</p> <p>chgrp() Changes the file group</p> <p>chmod() Changes the file mode</p> <p>chown() Changes the file owner</p> <p>clearstatcache() Clears the file status</p>

	<code>cache</code>
Filter	<p>PHP filters are used to validate and sanitize external input.</p> <p>The PHP filter extension has many of the functions needed for checking user input, and is designed to make data validation easier and quicker.</p> <p>Example:</p> <p>filter_has_var() Checks whether a variable of a specified input type exist</p> <p>filter_id() Returns the filter ID of a specified filter name</p> <p>filter_input() Gets an external variable (e.g. from form input) and optionally filters it</p> <p>filter_input_array() Gets external variables (e.g. from form input) and optionally filters them</p> <p>filter_list() Returns a list of all supported filter names</p>
FTP	<p>PHP filters are used to validate and sanitize external input.</p> <p>The PHP filter extension has many of the functions needed for checking user input, and is designed to make data validation easier and quicker.</p> <p>Example:</p> <p>ftp_alloc() Allocates space for a file to be uploaded to the FTP server</p> <p>ftp_cdup() Changes to the parent directory on the FTP server</p> <p>ftp_chdir() Changes the current directory on the FTP server</p> <p>ftp_chmod() Sets permissions on a file via FTP</p> <p>ftp_close() Closes an FTP connection</p>
Libxml	<p>The libxml functions and constants are used together with SimpleXML, XSLT and DOM functions.</p> <p>Example:</p> <p>libxml_clear_errors() Clears the libxml</p>

	<p>error buffer</p> <p>libxml_disable_entity_loader() Enables the ability to load external entities</p> <p>libxml_get_errors() Gets the errors from the the libxml error buffer</p> <p>libxml_get_last_error() Gets the last error from the the libxml error buffer</p> <p>libxml_set_external_entity_loader() Changes the default external entity loader</p>
Mail	<p>The mail() function is used to send a mail.</p> <p>Example:</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>Example:</p> <p>abs() Returns the absolute (positive) value of a number</p> <p>acos() Returns the arc cosine of a number</p> <p>acosh() Returns the inverse hyperbolic cosine of a number</p> <p>asin() Returns the arc sine of a number</p> <p>atan() Returns the arc tangent of a number in radians</p>
Misc	<p>The misc. functions were only placed here because none of the other categories seemed to fit.</p> <p>Example:</p> <p>connection_aborted() Checks whether the client has disconnected</p> <p>connection_status() Returns the current connection status</p> <p>connection_timeout() Deprecated from PHP 4.0.5. Checks whether the script has timed out</p> <p>constant() Returns the value of a constant</p> <p>define() Defines a constant</p>
MySQLi	<p>The MySQLi functions allows you to access MySQL database servers.</p> <p>Example:</p> <p>affected_rows() Returns the number of affected rows in the previous MySQL operation</p> <p>autocommit() Turns on or off auto-committing database modifications</p> <p>begin_transaction() Starts a transaction</p> <p>change_user() Changes the user of the specified database connection</p> <p>character_set_name() Returns the default character set for the database connection</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>Example:</p> <p>checkdnsrr() Checks DNS records for <i>type</i> corresponding to <i>host</i></p> <p>closelog() Closes the connection of system logger</p> <p>define_syslog_variables() Deprecated and removed in PHP 5.4. Initializes the variables used in syslog functions</p> <p>dns_check_record() Alias of checkdnsrr()</p> <p>dns_get_mx() Alias of getmxrr()</p>
SimpleXML	<p>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.</p> <p>Example:</p> <p>__construct() Creates a new SimpleXMLElement object</p> <p>__toString() Returns the string content of an element</p> <p>addAttribute() Appends an attribute to the SimpleXML element</p> <p>addChild() Appends a child element the SimpleXML element</p> <p>asXML() Returns a well-formed XML string (XML version 1.0) from a SimpleXML object</p>
Stream	<p>The Stream functions</p> <p>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.</p> <p>A wrapper is additional code which tells the stream how to handle specific protocols/encodings.</p> <p>Example:</p>

	<code>set_socket_blocking()</code> , <code>stream_bucket_prepend()</code> , <code>stream_context_create()</code> , <code>stream_context_get_default()</code> , <code>stream_context_get_options()</code>
String	<p>A string is a sequence of characters, like "Hello world!".</p> <p>Example:</p> <p>addslashes() Returns a string with backslashes in front of the specified characters</p> <p>addslashes() , bin2hex() , chop(),chr(),</p>
XML Parser	<p>The XML functions lets you parse, but not validate, XML documents.</p> <p>XML is a data format for standardized structured document exchange.</p> <p>Example:</p> <p>utf8_decode() Decodes an UTF-8 string to ISO-8859-1</p> <p>utf8_encode() Encodes an ISO-8859-1 string to UTF-8</p> <p>xml_error_string() Returns an error string from the XML parser</p> <p>xml_get_current_byte_index() Returns the current byte index from the XML parser</p> <p>xml_get_current_column_number() Returns the current column number from the XML parser</p>
Zip	<p>The Zip files functions allows you to read ZIP files.</p> <p>Example:</p> <p>zip_close() Closes a ZIP file archive</p> <p>zip_entry_close() Closes a ZIP directory entry</p> <p>zip_entry_compressedsize() Returns the compressed file size of a ZIP directory entry</p> <p>zip_entry_compressionmethod() Returns the compression method of a ZIP directory entry</p> <p>zip_entry_filesize() Returns the actual file size of a ZIP directory entry</p>
Timezones	<p>The <code>date_default_timezone_set()</code> function is an inbuilt function in PHP which is used to set the default timezone used by all date/time</p>

	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.

Example syntax in php:

```
<?php
```

```
// Declare a regular expression
$regex = '/^[a-zA-Z ]*$/';

// Declare a string
$nameString = 'Sharukh khan';

// Use preg_match() function to
// search string pattern
if(preg_match($regex, $nameString)) {
    echo("Name string matching with"
        . " regular expression");
}
else {
    echo("Only letters and white space"
        . " allowed in name string");
}

?>
```

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
$str1 = 'The quick brown fox.';
if (strpos($str1,'fox') !== false)
{
    echo 'fox is found the string';
}
else
{
    echo 'fox is not found 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  
  
$txt = "The quick brown fox";  
$str= preg_replace('/\W\w+\s*(\W*)$/','',$txt);  
echo $str  
  
?>
```

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

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

Expected output: 123,34.00

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

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

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

Expected output: Fox

```
<?php  
$sampletxt= 'The quick brown (fox)';  
preg_match('#\((.*?)\)#', $sampletxt, $match);  
print $match[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

```
<?php  
  
$string = 'abcde$ddfd @abcd )der]';  
$new = preg_replace("/[^a-zA-Z0-9 ]/","",$string);  
echo $new;  
  
?>
```

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.");} ?>
```