

# ACTIVITY ANSWER SHEET

|          |                 |
|----------|-----------------|
| Name     | Rhea Fe Cahanap |
| Section: | R2              |

- 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) {<br/>    // code to execute if condition is met<br/>}</pre>  |
| 2. if...else           | <pre>if (condition) {<br/>    // code to execute if condition is met }<br/>else {<br/>    // code to execute if condition is not met }</pre>  |
| 3. if...else if...else | <pre>if (condition) { // code to execute if condition is met }<br/>elseif (condition) { // code to execute if this condition is met }<br/>else { // code to execute if none of the conditions are met }</pre>                             |
| 4. switch...case       | <pre>switch (n) { case x: code to execute if n=x; break; case y: code to execute if<br/>n=y; break; case z: code to execute if n=z; break; // add more cases as<br/>needed default: code to execute if n is neither of the above; }</pre> |
| 5. for loop            | <pre>for (starting counter value; ending counter value; increment by which to<br/>increase) { // code to execute goes here }</pre>  |
| 6. do while loop       | <pre>do { // code to execute goes here; }<br/>while (condition that must apply);</pre>  |
| 7. while loop          | <pre>while (condition that must apply) {<br/>    // code to execute goes here }</pre>   |
| 8. foreach loop        | <pre>foreach (\$InsertYourArrayName as \$value) {<br/>    // code to execute goes here}</pre>   |
| 9. break statement     | <pre>break;</pre>   |
| 10. continue statement | <pre>continue;</pre>  |
| 11. try...catch        | <pre>try {<br/>    // run your code here<br/>}<br/>catch (exception ) {<br/>    //code to handle the exception<br/>}</pre>  |

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
$string = 1;
if(gettype($string) == gettype("")){
    echo "Not A number";
}
else { echo "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
$value = 1;
if($value%2)
{ if($value >= 0){ echo 'Positive and Odd'; }
  else { echo 'Negative and Odd';}}
else{
  if($value >= 0){ echo 'Positive and Even'; }
  else { echo 'Negative and Even';}}
?>
```

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
$string = 'Anna';
$stringLReverse = strrev($string);

if(strtoupper($string) == strtoupper($stringLReverse)){
    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
$string = 4;
$factorial = 1;
for ($x=$ string; $x>=1; $x--)
{
    $factorial = $factorial * $x;
}
echo "$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
```

```
<?php
$string = "ASA";
$strings = "The quick brown fox jumps over the lazy dog";
if(strpos(strtoupper($strings), strtoupper($string)) !== false){
    echo "$string is found the string";
} else{
    echo "$string is not found the string";
}
?>
```

Activity 2: PHP Built-in Functions

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

|             |   |
|-------------|---|
| Array       | An array is a special variable, which can hold more than one value at a time.   |
| Calendar    | Calendar extension contains function that simplifies converting between different calendar formats.   |
| Date        | Date/time functions allow you to get the date and time from the server where your PHP script runs. You can then use the date/time functions to format the date and time in several ways.  |
| Directory   | The directory functions allow you to retrieve information about directories and their contents.   |
| Error       | <p>The error functions are used to deal with error handling and logging.</p> <p>The error functions allow us to define own error handling rules, and modify the way the errors can be logged.</p> <p>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>   |
| File System | The file system functions allow you to access and manipulate the file system.   |
| Filter      | PHP filter is used to validate and filter data coming from insecure sources, like user input.   |
| FTP         | <p>The FTP functions give client access to file servers through the File Transfer Protocol (FTP).</p> <p>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. Not all of the FTP functions will work with every server or return the same results. The FTP functions became available with PHP 3.</p> <p>If you only wish to read from or write to a file on an FTP server, consider using the ftp:// wrapper with the File system functions which provide a simpler and more intuitive interface.</p> |

|            |  |
|------------|--|
| Libxml     | libxml functions and constants are used together with SimpleXML, XSLT and DOM functions.   |
| Mail       | The mail() function allows you to send emails directly from a script.  |
| Math       | The math functions can handle values within the range of integer and float types.  |
| Misc       | The misc. functions were only placed here because none of the other categories seemed to fit.  |
| MySQLi     | The MySQLi function allows you to access MySQL database servers.   |
| Network    | The Network function contains various network functions and let you manipulate information sent to the browser by the Web server, before any other output has been sent.   |
| 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.   |
| Stream     | 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.   |
| String     | The PHP string functions are part of the PHP core. No installation is required to use these functions.   |
| XML Parser | The XML functions lets you parse, but not validate, XML documents. XML is a data format for standardized structured document exchange. More information on XML can be found in our XML Tutorial. This extension uses the Expat XML parser. Expat is an event-based parser, it views an XML document as a series of events. When an event occurs, it calls a specified function to handle it. Expat is a non-validating parser, and ignores any DTDs linked to a document. However, if the document is not well formed it will end with an error message. Because it is an event-based, non |

|           |  |
|-----------|--|
|           | validating parser, Expat is fast and well suited for web applications.<br>The XML parser functions lets you create XML parsers and define handlers for XML events. |
| Zip       | The Zip files functions allows you to read ZIP files.  |
| Timezones |  |

### Activity 3: Regular Expression

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

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 = "Fox";
$strings = "The quick brown fox";

// Test if string contains the word
if(strpos(strtoupper($strings), strtoupper($string)) !== false){
    echo "$string is found the string";
} else{
    echo "$string 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
$string = 'The quick brown fox';
$removeLast = explode( " ", $string );
array_splice( $removeLast, -1 );

echo implode( " ", $removeLast );
?>
```

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

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

Expected output: 123,34.00

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

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

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

Expected output: Fox

```
<?php
$string = 'The quick brown (fox)';
preg_match('#\((.*?)\)#', $string, $match);
echo $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: abcdeddf d abcd der

```
<?php
$string="abcde$ddfd @abcd )der1]";
echo preg_replace("/[^A-Za-z0-9 ]/", "", $string);
?>
```

## Activity 4: Error Handling

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

Error Functions:

debug\_backtrace() - Used to generate a backtrace.  
debug\_print\_backtrace() - Prints a backtrace.  
error\_get\_last() - Gets the last error that occurred.  
error\_log() - Sends an error message to the web server's log, a file or a mail account.  
error\_reporting() - Specifies which PHP errors are reported.  
restore\_error\_handler() - Reverts to the previous error handler function.  
restore\_exception\_handler() - Goes back to the previous exception handler.  
set\_error\_handler() - Sets a user-defined function to handle script errors.  
set\_exception\_handler() - Sets an exception handler function defined by the user.  
trigger\_error() - Generates a user-level error message, you can also use user\_error().

Error Constants:

E\_ERROR - Fatal run-time errors that cause the halting of the script and can't be recovered from.  
E\_WARNING - Non-fatal run-time errors, execution of the script continues.  
E\_PARSE - Compile-time parse errors, should only be generated by the parser.  
E\_NOTICE - Run-time notices that indicate a possible error.  
E\_CORE\_ERROR - Fatal errors at PHP initialization, like an E\_ERROR in PHP core.  
E\_CORE\_WARNING - Non-fatal errors at PHP startup, similar to E\_WARNING but in PHP core.  
E\_COMPILE\_ERROR - Fatal compile-time errors generated by the Zend Scripting Engine.  
E\_COMPILE\_WARNING - Non-fatal compile-time errors by the Zend Scripting Engine.  
E\_USER\_ERROR - Fatal user-generated error, set by the programmer using trigger\_error().  
E\_USER\_WARNING - Non-fatal user-generated warning.  
E\_USER\_NOTICE - User-generated notice by trigger\_error().  
E\_STRICT - Suggestions by PHP to improve your code (needs to be enabled).  
E\_RECOVERABLE\_ERROR - Catchable fatal error caught by a user-defined handle.  
E\_DEPRECATED - Enable this to receive warnings about a code which is not futureproof.  
E\_USER\_DEPRECATED - User-generated warning for deprecated code.  
E\_ALL - All errors and warnings except E\_STRICT.