ACTIVITY ANSWER SHEET

Name	Jinomel L. Pajardo
Section:	BS-IT 3R2

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 execute if condition is met
	}

2. if…else	if (condition) {
	// code to execute if condition is met
	}
	else {
	// code to execute if condition is not met
	}

3. if…else if…else	if (condition) {
	//code to execute if condition is met
	}
	else if (condition) {
	// code to execute if this condition is met
	}
	else {
	// code to execute if none of the conditions are met
	}

4. switch…case	switch (n) {
	case x:
	code to execute if n=x;
	break;
	case y:
	code to execute if n=y;
	break;
	case z:
	code to execute if n=z;
	break;
	// add more cases as needed
	default:
	code to execute if n is neither of the above;
	}

5. for loop	for (starting counter value; ending counter value; increment by which to increase) {
	// code to execute goes here
	}

6. do while loop	do {
	// code to execute goes here;
	} while (condition that must apply);

7. while loop	while (condition that must apply) {
	// code to execute goes here
	}

8. foreach loop	foreach ($InsertYourArrayName as $value) {
	// code to execute goes here
	}

9. break statement	break;
10. continue statement	continue;

11. try…catch	try {
	  //A function using an exception should be in a "try" block. If the exception does not trigger, the code will continue as normal.
	}
	
	//catch exception
	catch(Exceptions) {
	  //code of exceptions
	}

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	The calendar extension contains functions that simplifies converting between different calendar formats.

Date	The 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	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.
File System	The filesystem functions allow you to access and manipulate the filesystem.
Filter	This PHP filters is used to validate and filter data coming from insecure sources, like user input.
FTP	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. Not all of the FTP functions will work with every server or return the same results. The FTP functions became available with PHP 3. If you only wish to read from or write to a file on an FTP server, consider using the ftp:// wrapper with the Filesystem functions which provide a simpler and more intuitive interface.
Libxml	The 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 functions allows you to access MySQL database servers.
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.
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. 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	Below is a complete list of the timezones supported by PHP, which are useful with several PHP date functions.

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";

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: abcdeddfd 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.

PHP 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.