ACTIVITY ANSWER SHEET

Name	Karl Malone P. Obsioma
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	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	1. jump statement; 2. break;

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

11. try…catch	try {
	print "this is our try block n";
	throw new Exception();
	}
	catch (Exception $e) {
	print "Somehing went wrong! n";

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
 if ( (int) '1' !== 1 )
{
     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
function check($number){
     if($number % 2 == 0){
         echo "Even ";
       }
     else{
         echo "Odd ";
     }
 }
   function sample($number){
     if($number >= 0){
         echo "& Positive";
       }
     else{
        echo "& Negative";
     }
 } $number = -1; check($number); sample($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
 function Palindrome($string){
  if (strrev($string) == $string){
   return 1;
  }
  else{
   return 0;  }}
// Driver Code $original = "anna";
 if(Palindrome($original)){
  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

```
<?php
$n = 4;
 $x = 1;
 for($i=1;$i<=$n-1;$i++)
{
  $x*=($i+1);
 }
 echo "Factorial of $n is = $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
 $n = 3;
echo "sample input = " . $n . "\n";
 $count = 1;
for ($i = $n; $i > 0; $i--) {
   for ($j = $i; $j < $n + 1; $j++)
   {
      printf("%4s", $count);
      $count++;
    }
  echo "\n";
    }
 ?>
```

Activity 2: PHP Built-in Functions

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

Array	array() array_change_key_case()
	array_chunk() array_column()
	array_combine() array_count_values()
	array_diff() array_diff_assoc()
	array_diff_key() array_diff_uassoc()
	array_diff_ukey() array_fill()
	array_fill_keys() array_filter() array_flip()
	array_intersect() array_intersect_assoc()

	array_intersect_key() array_intersect_uassoc() array_intersect_ukey() array_key_exists() array_keys() array_map() array_merge() array_merge_recursive() array_multisort() array_pad() array_pop() array_product() array_push() array_rand() array_reduce() array_replace() array_replace_recursive() array_reverse() array_search() array_shift() array_slice() array_splice() array_sum() array_udiff() array_udiff_assoc() array_udiff_uassoc() array_uintersect() array_uintersect_assoc() array_uintersect_uassoc() array_unique() array_unshift() array_values() array_walk() array_walk_recursive() arsort() asort() compact() count() current() each() end() extract() in_array() key() krsort() ksort() list() natcasesort() natsort() next() pos() prev() range() reset() rsort() shuffle() sizeof() sort() uasort() uksort() usort()
Calendar	cal_days_in_month() cal_from_jd() cal_info() cal_to_jd() easter_date() easter_days() frenchtojd() gregoriantojd() jddayofweek() jdmonthname() jdtofrench() jdtogregorian() jdtojewish() jdtojulian() jdtounix() jewishtojd() juliantojd() unixtojd()
Date	checkdate() date_add() date_create_from_format() date_create() date_date_set() date_default_timezone_get() date_default_timezone_set() date_diff() date_format() date_get_last_errors() date_interval_create_from_date_string() date_interval_format() date_isodate_set() date_modify() date_offset_get() date_parse_from_format() date_parse() date_sub() date_sun_info() date_sunrise() date_sunset() date_time_set() date_timestamp_get() date_timestamp_set() date_timezone_get() date_timezone_set() date() getdate() gettimeofday() gmdate() gmmktime() gmstrftime() idate() localtime() microtime() mktime() strftime() strptime() strtotime() time() timezone_abbreviations_list() timezone_identifiers_list() timezone_location_get() timezone_name_from_ abbr() timezone_name_get() timezone_offset_get() timezone_open() timezone_transitions_get() timezone_version_get()
Directory	chdir() chroot() closedir() dir() getcwd() opendir() readdir() rewinddir() scandir()
Error	debug_backtrace() debug_print_backtrace() error_clear_last() error_get_last() error_log() error_reporting() restore_error_handler()

	restore_exception_handler() set_error_handler() set_exception_handler() trigger_error() user_error()
File System	basename() chgrp() chmod() chown() clearstatcache() copy() delete() dirname() disk_free_space() disk_total_space() diskfreespace() fclose() feof() fflush() fgetc() fgetcsv() fgets() fgetss() file() file_exists() file_get_contents() file_put_contents() fileatime() filectime() filegroup() fileinode() filemtime() fileowner() fileperms() filesize() filetype() flock() fnmatch() fopen() fpassthru() fputcsv() fputs() fread() fscanf() fseek() fstat() ftell() ftruncate() fwrite() glob() is_dir() is_executable() is_file() is_link() is_readable() is_uploaded_file() is_writable() is_writeable() lchgrp() lchown() link() linkinfo() lstat() mkdir() move_uploaded_file() parse_ini_file() parse_ini_string() pathinfo() pclose() popen() readfile() readlink() realpath() realpath_cache_get() realpath_cache_size() rename() rewind() rmdir() set_file_buffer() stat() symlink() tempnam() tmpfile() touch() umask() unlink()

Filter	filter_has_var()
	filter_id()
	filter_input()
	filter_input_array()
	filter_list()
	filter_var()
	filter_var_array()

FTP	ftp_alloc() ftp_cdup() ftp_chdir() ftp_chmod() ftp_close() ftp_connect() ftp_delete() ftp_exec() ftp_fget() ftp_fput() ftp_get() ftp_get_option() ftp_login() ftp_mdtm() ftp_mkdir() ftp_mlsd() ftp_nb_continue() ftp_nb_fget() ftp_nb_fput() ftp_nb_get() ftp_nb_put() ftp_nlist() ftp_pasv() ftp_put() ftp_pwd() ftp_quit() ftp_raw() ftp_rawlist() ftp_rename() ftp_rmdir() ftp_set_option() ftp_site() ftp_size() ftp_ssl_connect() ftp_systype()

Libxml	libxml_clear_errors()
	libxml_disable_entity_loader()
	libxml_get_errors() libxml_get_last_error()
	libxml_set_external_entity_loader()
	libxml_set_streams_context()
	libxml_use_internal_errors()

Mail	ezmlm_hash() mail()
Math	abs() acos() acosh() asin() asinh() atan() atan2() atanh() base_convert() bindec() ceil() cos() cosh() decbin() dechex() decoct() deg2rad() exp()

	expm1() floor() fmod() getrandmax() hexdec() hypot() intdiv() is_finite() is_infinite() is_nan() lcg_value() log() log10() log1p() max() min() mt_getrandmax() mt_rand() mt_srand() octdec() pi() pow() rad2deg() rand() round() sin() sinh() sqrt() srand() tan() tanh()
Misc	connection_aborted() connection_status() connection_timeout() constant() define() defined() die() eval() exit() get_browser() __halt_compiler() highlight_file() highlight_string() hrtime() ignore_user_abort() pack() php_strip_whitespace() show_source() sleep() sys_getloadavg() time_nanosleep() time_sleep_until() uniqid() unpack() usleep()
MySQLi	affected_rows() autocommit() begin_transaction() change_user() character_set_name() close() commit() connect() connect_errno() connect_error() data_seek() debug() dump_debug_info() errno() error() error_list() fetch_all() fetch_array() fetch_assoc() fetch_field() fetch_field_direct() fetch_fields() fetch_lengths() fetch_object() fetch_row() field_count() affected_rows() autocommit() begin_transaction() change_user() character_set_name() close() commit() connect() connect_errno() connect_error() data_seek() debug() dump_debug_info() errno() error() error_list() fetch_all() fetch_array() fetch_assoc() fetch_field() fetch_field_direct() fetch_fields() fetch_lengths() fetch_object() fetch_row() field_count() field_seek() get_charset() get_client_info() get_client_stats() get_client_version() get_connection_stats() get_host_info() get_proto_info() get_server_info() get_server_version() info() init() insert_id() kill() more_results() multi_query() next_result() options() ping() poll() prepare() query() real_connect() real_escape_string() real_query() reap_async_query() refresh() rollback() select_db() set_charset() set_local_infile_default() set_local_infile_handler() sqlstate() ssl_set() stat() stmt_init() store_result() thread_id() thread_safe() use_result() warning_count()
Network	checkdnsrr() closelog() define_syslog_variables() dns_check_record() dns_get_mx() dns_get_record() fsockopen() gethostbyaddr() gethostbyname() gethostbynamel() gethostname() getmxrr() getprotobyname()

	getprotobynumber() getservbyname() getservbyport() header_register_callback() header_remove() header() headers_list() headers_sent() http_response_code() inet_ntop() inet_pton() ip2long() long2ip() openlog() pfsockopen() setcookie() setrawcookie() socket_get_status() socket_set_blocking() socket_set_timeout() syslog()
SimpleXML	_construct() _toString() addAttribute() addChild() asXML() attributes() children() count() getDocNamespaces() getName() getNamespaces() registerXPathNamespace() saveXML() simplexml_import_dom() simplexml_load_file() simplexml_load_string() xpath() current() getChildren() hasChildren() key() next() rewind() valid()
Stream	set_socket_blocking() stream_bucket_prepend() stream_context_create() stream_context_get_default() stream_context_get_options() stream_context_get_params() stream_context_set_default() stream_context_set_options() stream_context_set_params() stream_copy_to_stream() stream_filter_append() stream_filter_prepend() stream_filter_register() stream_filter_remove() stream_get_contents() stream_get_filters() stream_get_line() stream_get_meta_data() stream_get_transports() stream_get_wrappers() stream_is_local() stream_isatty() stream_notification_callback() stream_register_wrapper() stream_resolve_include_path() stream_select() stream_set_blocking() stream_set_chunk_size() stream_set_read_buffer() stream_set_timeout() stream_set_write_buffer() stream_socket_accept() stream_socket_client() stream_socket_enable_crypto() stream_socket_get_name() stream_socket_pair() stream_socket_recvfrom() stream_socket_sendto() stream_socket_server() stream_socket_shutdown() stream_supports_lock() stream_wrapper_register() stream_wrapper_restore() stream_wrapper_unregister()

String	addcslashes() addslashes() bin2hex() chop() chr() chunk_split() convert_cyr_string() convert_uudecode() convert_uuencode() count_chars() crc32() crypt() echo() explode() fprintf() get_html_translation_table() hebrev() hebrevc() hex2bin() html_entity_decode() htmlentities() htmlspecialchars_decode() htmlspecialchars() implode() join() lcfirst() levenshtein() localeconv() ltrim() md5() md5_file() metaphone() money_format() nl_langinfo() nl2br() Inserts number_format() ord() parse_str() print() printf() quoted_printable_decode() quoted_printable_encode() quotemeta() rtrim() setlocale() sha1() sha1_file() similar_text() soundex() sprintf() sscanf() str_getcsv() str_ireplace() str_pad() str_repeat() str_replace() str_rot13() str_shuffle() str_split() str_word_count() strcasecmp() strchr() strcmp() strcoll() strcspn() strip_tags() stripcslashes() stripslashes() stripos() stristr() strlen() strnatcasecmp() strnatcmp() strncasecmp() strncmp() strpbrk() strpos() strrchr() strrev() strripos() strrpos() strspn() strstr() strtok() strtolower() strtoupper() strtr() substr() substr_compare() substr_count() substr_replace() trim() ucfirst() ucwords() vfprintf() vprintf() vsprintf() wordwrap()

XML Parser	utf8_decode()
	utf8_encode() xml_error_string()
	xml_get_current_byte_index()
	xml_get_current_column_number()
	xml_get_current_line_number()
	xml_get_error_code()
	xml_parse() xml_parse_into_struct()
	xml_parser_create_ns()
	xml_parser_create()
	xml_parser_free()
	xml_parser_get_option()
	xml_parser_set_option()
	xml_set_character_data_handler()
	xml_set_default_handler()
	xml_set_element_handler()
	xml_set_external_entity_ref_handler()
	xml_set_notation_decl_handler()
	xml_set_object()
	xml_set_processing_instruction_handler()
	xml_set_start_namespace_decl_handler ()
	xml_set_unparsed_entity_decl_handler()

Zip	zip_close()
	zip_entry_close()
	zip_entry_compressedsize()
	zip_entry_compressionmethod()
	zip_entry_filesize()
	zip_entry_name()
	zip_entry_open()
	zip_entry_read()
	zip_open()

	zip_read()
Timezones	Africa America Antartica Arctic Asia Atlantic Australia Europe Indian Pacific

Activity 3: Regular Expression

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

```
<?php
 $pattern = "/ca[kf]e/";
$text = "He was singing in a bar.";
 if(preg_match($pattern, $text)){
 echo "Match found!";
 }
 else{ echo "Match not found."; }
 ?>
```

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
$pattern = '/[^\w]fox\s/';
 if (preg_match($pattern, 'The quick brown fox'))
{
echo "Fox doesn't found the string"."\n";
}   else
 echo "Fox is found the string"."\n";
?>
```

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
$str1 = 'The quick brown fox';
echo preg_replace('/\W\w+\s*(\W*)$/', '$1', $str1)."\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
$str1 = "$12,334.00A#";
echo preg_replace("/[^0-9,.]/", "", $str1)."\n";
?>
```

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

```
<?php
$str1 = "$12,334.00A#";
echo preg_replace("/[^0-9,.]/", "", $str1)."\n";
?>
 <?php
$my_text = 'The quick brown [Fox].';
 preg_match('#\[(.*?)\]#', $my_text, $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
$string = 'abcde$ddfd@abcd )der]';
$newstr = preg_replace("/[^A-Za-z0-9 ]/", '', $string);
echo ''.$newstr."\n";
?>
```

Activity 4: Error Handling

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

- 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)

- 1. Notices:

These are small, non-critical errors that PHP encounters while executing a script - for example, accessing a variable that has not yet been defined. By default, such errors are not displayed to the user at all - although the default behavior can be changed.

2. Warnings:

Warnings are more severe errors like attempting to include() a file which does not exist. By default, these errors are displayed to the user, but they do not result in script termination.

3. Fatal errors:

These are critical errors - for example, instantiating an object of a non-existent class, or calling a non-existent function. These errors cause the immediate termination of the script, and PHP's default behavior is to display them to the user when they take place