**ACTIVITY ANSWER SHEET**

|  |  |
| --- | --- |
| Name | Villar, Richie Ann Lourene |
| Section: | BSIT-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 | 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);   1. While loop   while (condition is true) {     code to be executed; } |
| 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) {  //code to handle the exception  }  finally {  //optional code that always runs  } |

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  $t = '1';  if (is\_integer($t))  {  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  $t = -1;  if($t >= 0 )  {  if($t%2 == 0)  {  echo "positive and even";  }  else  {  echo "positive and odd";  }  }  else if ($t < 0)  {  if($t%2 == 0)  {  echo "negative and even";  }else  {  echo "negative 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  $string = "Katniss";  //code for palindrom  if(strrev($string) == $string)  {  echo "Palindrome";  }  else  {  echo "Not a palindrom";  }  ?> |

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

Sample input: 4

Expected output: 24

|  |
| --- |
| <?php  $number = 4;  $increment = 1;  $fact = 1;  //code for factorial    for($increment = 1;$increment<$number;$increment++)  {  $fact = $fact \* ($increment+1);  }  echo $fact;  ?> |

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  $lineNum = 10;  $count = 1;  $s = 0;  $i =0;  //floyd triangle  for($i = $lineNum;$i > 0;$i--)  {  for($j = $i; $j < $lineNum;$j++)  {  printf("%4s", $count);  $count++;  }  echo nl2br("\n");  }  ?> |

**Activity 2: PHP Built-in Functions**

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

|  |  |
| --- | --- |
| Array | The array functions allow you to access and manipulate arrays.  array\_fill() Fills an array with values  array\_fill\_keys() Fills an array with values, specifying keys  array\_filter() Filters the values of an array using a callback function  array\_flip() Flips/Exchanges all keys with their associated values in an array  array\_intersect() Compare arrays, and returns the matches (compare values only) |
| Calendar | The calendar extension contains functions that simplifies converting between different calendar formats.  cal\_days\_in\_month() Returns the number of days in a month for a specified year and calendar  easter\_days() Returns the number of days after March 21, that the Easter Day is in a specified year  frenchtojd() Converts a French Republican date to a Julian Day Count  gregoriantojd() Converts a Gregorian date to a Julian Day Count  jddayofweek() Returns the day of the week |
| 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.  checkdate() Validates a Gregorian date  date\_add() Adds days, months, years, hours, minutes, and seconds to a date  date\_create\_from\_format() Returns a new DateTime object formatted according to a specified format  date\_create() Returns a new DateTime object  date\_date\_set() Sets a new date |
| Directory | The directory functions allow you to retrieve information about directories and their contents.  getcwd() Returns the current working directory  opendir() Opens a directory handle  readdir() Returns an entry from a directory handle  rewinddir() Resets a directory handle  scandir() Returns an array of files and directories of a specified directory |
| Error | The error functions are used to deal with error handling and logging.  display\_startup\_errors "0"  log\_errors "0"  log\_errors\_max\_len "1024"  ignore\_repeated\_errors "0"  ignore\_repeated\_source "0" |
| File System | The filesystem functions allow you to access and manipulate the filesystem.  allow\_url\_fopen "1" Allows fopen()-type functions to work with URLs PHP\_INI\_SYSTEM  allow\_url\_include "0" (available since PHP 5.2) PHP\_INI\_SYSTEM  user\_agent NULL Defines the user agent for PHP to send (available since PHP 4.3) PHP\_INI\_ALL  default\_socket\_timeout "60" Sets the default timeout, in seconds, for socket based streams (available since PHP 4.3) PHP\_INI\_ALL  sys\_temp\_dir "" (available since PHP 5.5) PHP\_INI\_SYSTEM |
| Filter | This PHP filters is used to validate and filter data coming from insecure sources, like user input.  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 |
| FTP | The FTP functions give client access to file servers through the File Transfer Protocol (FTP).  ftp\_login() Logs in to the FTP connection  ftp\_mdtm() Returns the last modified time of a specified file  ftp\_mkdir() Creates a new directory on the FTP server  ftp\_mlsd() Returns the list of files in the specified directory  ftp\_nb\_continue() Continues retrieving/sending a file (non-blocking) |
| Libxml | The libxml functions and constants are used together with SimpleXML, XSLT and DOM functions.  libxml\_clear\_errors() Clears the libxml error buffer  libxml\_disable\_entity\_loader() Enables the ability to load external entities  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 |
| Mail | The mail() function allows you to send emails directly from a script.  ezmlm\_hash() Calculates the hash value needed by EZMLM  mail() Allows you to send emails directly from a script |
| Math | The math functions can handle values within the range of integer and float types.  decbin() Converts a decimal number to a binary number  dechex() Converts a decimal number to a hexadecimal number  decoct() Converts a decimal number to an octal number  deg2rad() Converts a degree value to a radian value  exp() Calculates the exponent of e |
| Misc | The math functions can handle values within the range of integer and float types.  defined() Checks whether a constant exists  die() Alias of exit()  eval() Evaluates a string as PHP code  exit() Prints a message and exits the current script  get\_browser() Returns the capabilities of the user's browser |
| MySQLi | The MySQLi functions allows you to access MySQL database servers.  errno() Returns the last error code for the most recent function call  error() Returns the last error description for the most recent function call  error\_list() Returns a list of errors for the most recent function call  fetch\_all() Fetches all result rows as an associative array, a numeric array, or both  fetch\_array() Fetches a result row as an associative, a numeric array, or both |
| 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.  getprotobyname() Returns the protocol number for a given protocol name  getprotobynumber() Returns the protocol name for a given protocol number  getservbyname() Returns the port number for a given Internet service and protocol  getservbyport() Returns the Internet service for a given port and protocol  header\_register\_callback() Calls a header function |
| SimpleXML | SimpleXML is an extension that allows us to easily manipulate and get XML data.  getDocNamespaces() Returns the namespaces declared in document  getName() Returns the name of an element  getNamespaces() Returns the namespaces used in document  registerXPathNamespace() Creates a namespace context for the next XPath query  saveXML() Alias of asXML() |
| Stream | A stream is a resource object which exhibits streamable behavior.  stream\_context\_get\_options()  stream\_context\_get\_params()  stream\_context\_set\_default()  stream\_context\_set\_options()  stream\_context\_set\_params() |
| String | The PHP string functions are part of the PHP core. No installation is required to use these functions.  crc32() Calculates a 32-bit CRC for a string  crypt() One-way string hashing  echo() Outputs one or more strings  explode() Breaks a string into an array  fprintf() Writes a formatted string to a specified output stream |
| XML Parser | The XML parser functions lets you create XML parsers and define handlers for XML events.  xml\_get\_current\_column\_number() Returns the current column number from the XML parser  xml\_get\_current\_line\_number() Returns the current line number from the XML parser  xml\_get\_error\_code() Returns an error code from the XML parser  xml\_parse() Parses an XML document  xml\_parse\_into\_struct() Parses XML data into an array |
| Zip | The Zip files functions allows you to read ZIP files.  zip\_entry\_name() Returns the name of a ZIP directory entry  zip\_entry\_open() Opens a directory entry in a ZIP file for reading  zip\_entry\_read() Reads from an open directory entry in the ZIP file  zip\_open() Opens a ZIP file archive  zip\_read() Reads the next file in a open ZIP file archive |
| Timezones | PHP Date/Time Functions  PHP gmdate() Function  PHP strtotime() Function  PHP Date and Time  PHP Tryit Editor v1.1 |

**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 = "The quick brown fox";  $test = "/Fox/i";  if (preg\_match($test, $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);

?>