DEPARTMENT OF COMPUTER STUDIES

**(Applications Development and Emerging Technologies)**

**PRE-SUMMATIVE ASSESSMENT**

2

**PHP OPERATORS AND CONTROL STRUCTURE**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Student Name / Group Name:** | **Patrick Andrew Tria** | | | |
| **Members (if Group):** |  | **Name** | **Role** |  |
|  |  |  |
|  |  |  |
|  |  |  |
| **Section:** | 1A TN31 | | | |
| **Professor:** | Mr. Abraham Magpantay | | | |

### PROGRAM OUTCOME/S (PO) ADDRESSED BY THE LABORATORY EXERCISE

* + Design, implement and evaluate computer-based systems or applications to meet desired needs and requirements.

### COURSE LEARNING OUTCOME/S (CLO) ADDRESSED BY THE LABORATORY EXERCISE

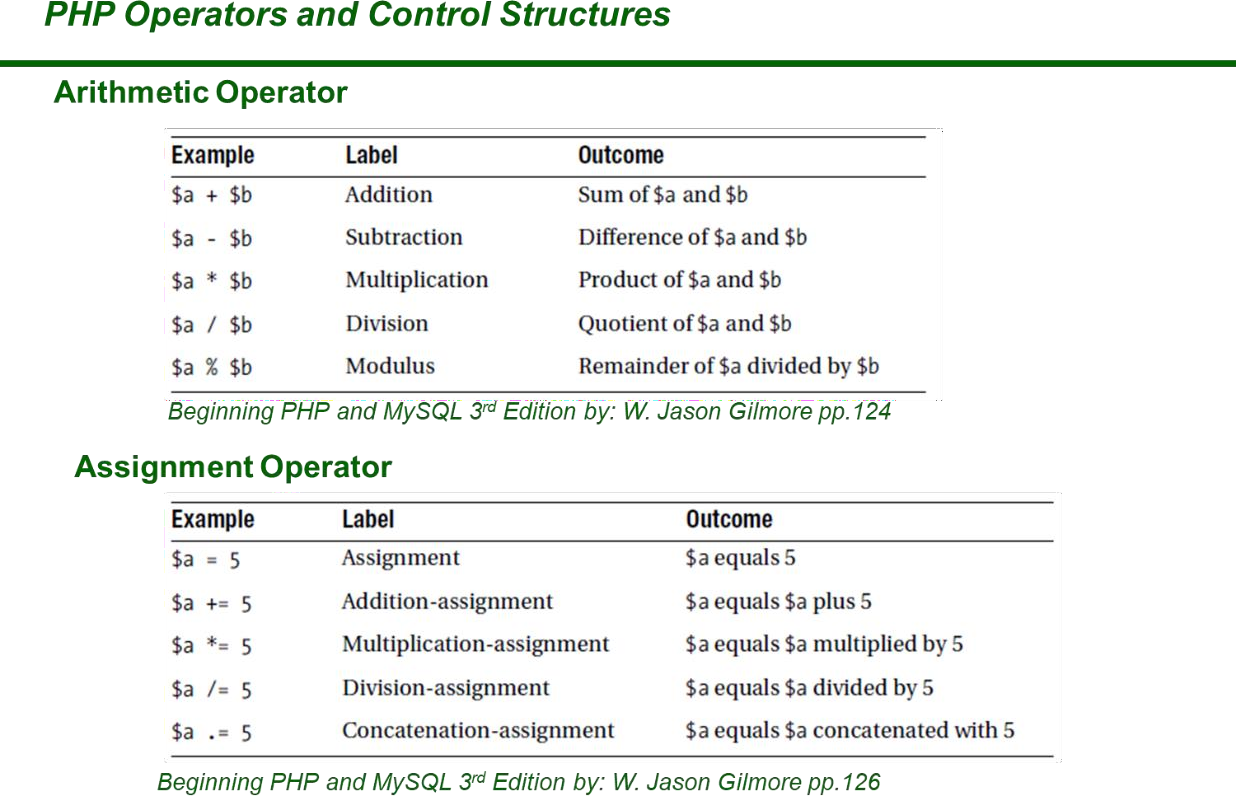
* + Understand and apply best practices and standards in the development of website.

### INTENDED LEARNING OUTCOME/S (ILO) OF THE LABORATORY EXERCISE

At the end of this exercise, students must be able to:

* + To understand the different types of operators that are available on PHP.
  + To know what is operator precedence and operator associativity in PHP.
  + To use escape sequence properly in the program.
  + To know the different approach of control structures.
  + To know the fundamentals syntax for conditional and looping structures.
  + To properly use the compound expression using the logical operators.
  + To know the rules of break, continue, and goto statements.

### BACKGROUND INFORMATION



Try following example to understand all the arithmetic operators.

<html>

<head>

<title>Arithmetical Operators</title>

</head>

<body>

<?php

$a = 42;

$b = 20;

$c = $a + $b;

echo "Addition Operation Result: $c <br/>";

$c = $a - $b;

echo "Subtraction Operation Result: $c <br/>";

$c = $a \* $b;

echo "Multiplication Operation Result: $c <br/>";

$c = $a / $b;

echo "Division Operation Result: $c <br/>";

$c = $a % $b;

echo "Modulus Operation Result: $c <br/>";

$c = $a++;

echo "Increment Operation Result: $c <br/>";

$c = $a--;

echo "Decrement Operation Result: $c <br/>";

?>

</body>

</html>

This will produce the following result –

Addition Operation Result: 62 Subtraction Operation Result: 22 Multiplication Operation Result: 840 Division Operation Result: 2.1 Modulus Operation Result: 2 Increment Operation Result: 42 Decrement Operation Result: 43

**PHP Conditional Statements**

Very often when you write code, you want to perform different actions for different conditions. You can use conditional statements in your code to do this.

In PHP we have the following conditional statements:

* + statement - executes some code if one condition is true

if...else

if

* + statement - executes some code if a condition is true and another code if that condition is false
  + if...elseif...else statement - executes different codes for more than two conditions
  + switch statement - selects one of many blocks of code to be executed

# PHP - The if Statement

The if statement executes some code if one condition is true.

## Syntax

if (*condition*) {

*code to be executed if condition is true*;

}

## Example

<?php

$t = date("H");

if ($t < "20") {

echo "Have a good day!";

}

?>

Output "Have a good day!" if the current time (HOUR) is less than 20:

# PHP – The if…else Statement

The if...else statement executes some code if a condition is true and another code if that condition is false.

## Syntax

if (*condition*) {

*code to be executed if condition is true;*

} else {

*code to be executed if condition is false;*

}

## Example

<?php

$t = date("H");

if ($t < "20") {

echo "Have a good day!";

} else {

echo "Have a good night!";

}

?>

Output "Have a good day!" if the current time is less than 20, and "Have a good night!" otherwise:

# PHP - The if...elseif...else Statement

The if...elseif...else statement executes different codes for more than two conditions.

## Syntax

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;*

}

## Example

Output "Have a good morning!" if the current time is less than 10, and "Have a good day!" if the current time is less than 20. Otherwise it will output "Have a good night!":

<?php

$t = date("H");

if ($t < "10") {

echo "Have a good morning!";

} elseif ($t < "20") { echo "Have a good day!";

} else {

echo "Have a good night!";

}

?>

# The PHP switch Statement

Use the switch statement to **select one of many blocks of code to be executed**.

## Syntax

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;*

}

## Example

<?php

$favcolor = "red";

switch ($favcolor) { case "red":

echo "Your favorite color is red!"; break;

case "blue":

echo "Your favorite color is blue!"; break;

case "green":

echo "Your favorite color is green!"; break;

default:

echo "Your favorite color is neither red, blue, nor green!";

}

?>

This is how it works: First we have a single expression *n* (most often a variable), that is evaluated once. The value of the expression is then compared with the values for each case in the structure. If there is a match, the block of

default

break

code associated with that case is executed. Use running into the next case automatically. The match is found.

# PHP Loops

to prevent the code from statement is used if no

Often when you write code, you want the same block of code to run over and over again a certain number of times. So, instead of adding several almost equal code-lines in a script, we can use loops.

Loops are used to execute the same block of code again and again, as long as a certain condition is true.

In PHP, we have the following loop types:

* + while - loops through a block of code as long as the specified condition is true
  + do...while - loops through a block of code once, and then repeats the loop as long as the specified condition is true
  + - loops through a block of code a specified number of times

foreach

for

* + - loops through a block of code for each element in an array

# PHP while Loop

The while loop - Loops through a block of code as long as the specified condition is true.

The while loop executes a block of code as long as the specified condition is true.

**Syntax**

while (*condition is true*) {

*code to be executed*;

}

## Example

The example below displays the numbers from 1 to 5:

<?php

$x = 1;

while($x <= 5) {

echo "The number is: $x <br>";

$x++;

}

?>

## Example Explained

* + $x = 1; - Initialize the loop counter ($x), and set the start value to 1
  + $x <= 5 - Continue the loop as long as $x is less than or equal to 5
  + $x++; - Increase the loop counter value by 1 for each iteration

# PHP do while Loop

The do...while loop - Loops through a block of code once, and then repeats the loop as long as the specified condition is true.

The do...while loop will always execute the block of code once, it will then check the condition, and repeat the loop while the specified condition is true.

**Syntax**

do {

*code to be executed;*

} while (*condition is true*);

## Example

The example below first sets a variable $x to 1 ($x = 1). Then, the do while loop will write some output, and then increment the variable $x with 1. Then the condition is checked (is $x less than, or equal to 5?), and the loop will continue to run as long as $x is less than, or equal to 5:

<?php

$x = 1;

do {

echo "The number is: $x <br>";

$x++;

} while ($x <= 5);

?>

**Note:** In a

loop the condition is tested AFTER executing the

statements within the loop. This means that the loop will execute its statements at least once, even if the condition is false. See example below.

do...while

do...while

# PHP for Loop

The for loop - Loops through a block of code a specified number of times.

The for loop is used when you know in advance how many times the script should run.

**Syntax**

for (*init counter; test counter; increment counter*) {

*code to be executed for each iteration;*

}

Parameters:

* + *init counter*: Initialize the loop counter value
  + *test counter*: Evaluated for each loop iteration. If it evaluates to TRUE, the loop continues. If it evaluates to FALSE, the loop ends.
  + *increment counter*: Increases the loop counter value

## Example

The example below displays the numbers from 0 to 10:

<?php

for ($x = 0; $x <= 10; $x++) { echo "The number is: $x <br>";

}

?>

# PHP foreach Loop

The foreach loop - Loops through a block of code for each element in an array.

The foreach loop works only on arrays, and is used to loop through each key/value pair in an array.

**Syntax**

foreach ($*array* as $*value*) {

*code to be executed;*

}

For every loop iteration, the value of the current array element is assigned to

$value and the array pointer is moved by one, until it reaches the last array element.

## Example

The example will output the values of the given array ($colors):

<?php

$colors = array("red", "green", "blue", "yellow");

foreach ($colors as $value) { echo "$value <br>";

}

?>

# PHP Break and Continue

You have already seen the break statement used in an earlier chapter of this tutorial. It was used to "jump out" of a switch statement.

# PHP Break

The break statement can also be used to jump out of a loop. This example jumps out of the loop when **x** is equal to **4**:

## Example

<?php

for ($x = 0; $x < 10; $x++) { if ($x == 4) {

break;

}

echo "The number is: $x <br>";

}

?>

# PHP Continue

The continue statement breaks one iteration (in the loop), if a specified condition occurs, and continues with the next iteration in the loop.

This example skips the value of **4**:

## Example

<?php

for ($x = 0; $x < 10; $x++) { if ($x == 4) {

continue;

}

echo "The number is: $x <br>";

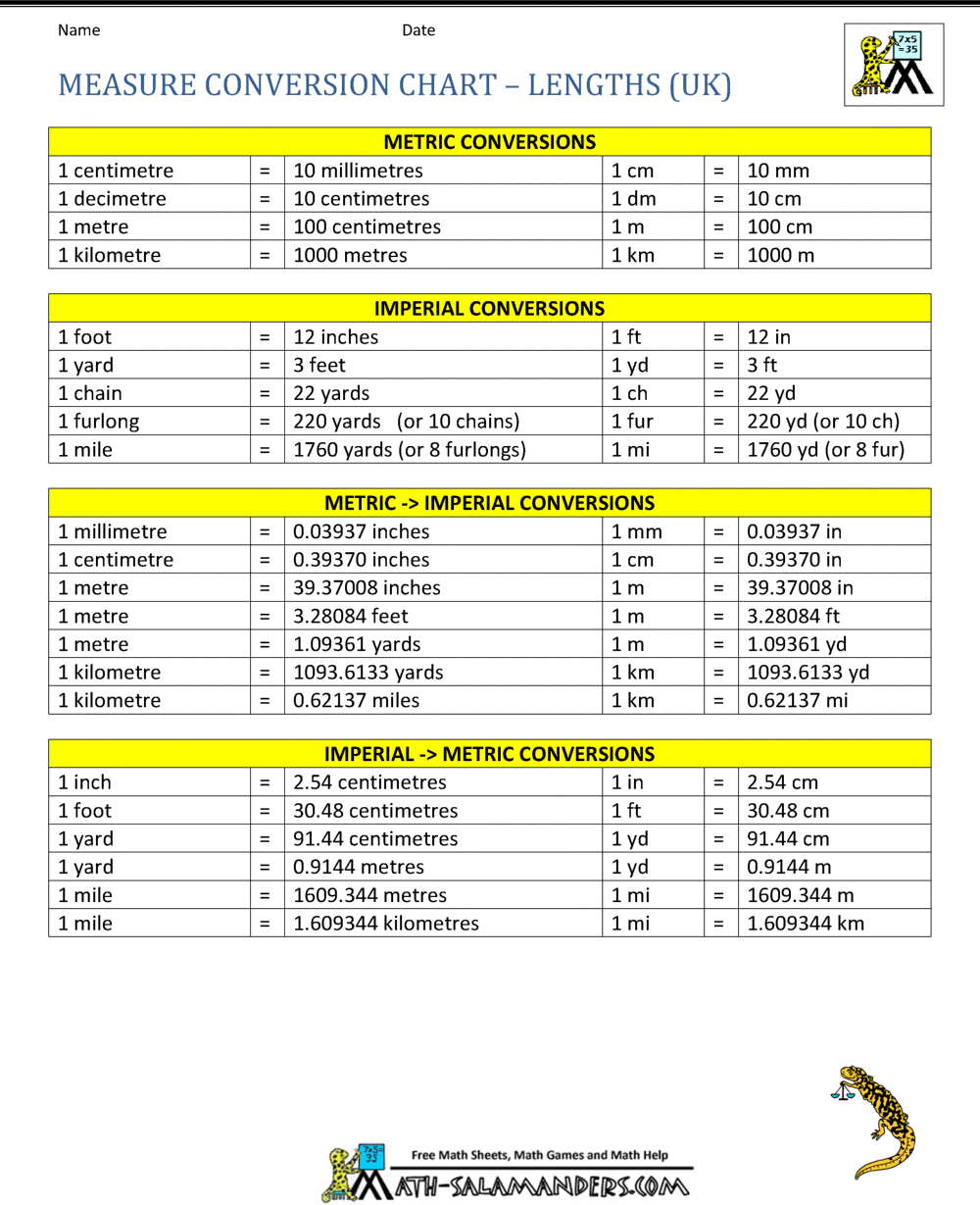
}

?>

### GRADING SYSTEM / RUBRIC (please see separate sheet)

1. **LABORATORY ACTIVITY**
2. Using PHP operators create a length conversion page, integrated with HTML and CSS (note: use formula for each conversion)

Example: 1 meter = 100 centimeter Sample Output:



1. Using conditional statement create a grade ranking program, integrated with HTML and CSS.

Example

Grade = 92 Ranking: A-

Use the equivalents below

A: 93-100

A-: 90-92

B+: 87-89

B: 83-86

B-: 80-82

C+: 77-79

C: 73-76

C-: 70-72

D+: 67-69

D: 63-66

D-: 60-62

F: Below 60 Sample Output

Grade: 95

Rank: A

Picture

Name: First Name MI. Lastame

1. Using Looping Statements write a program which will give you all of the potential combinations of a two-digit decimal combination, printed in a comma delimited format :

Sample output :

00, 01, 02, 03, 04, 05, 06, 07, 08, 09, 10, 11, 12, 13, 14, 15, 16, 17, 18,

19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37,

38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56,

57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75,

76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94,

95, 96, 97, 98, 99,

***Snip and paste your source codes here. Snip it directly from the IDE so that colors of the codes are preserved for readability. Include additional pages if necessary.***

### QUESTION AND ANSWER

1. **What are the different Arithmetic operators used in the lab activity? describe each**

The different Arithmetic Operators used are **+,** is used for addition. **-,** for subtraction, **\*** for multiplication and **/** for division. Operators like **++** are also used as increment and **–** for decrement. **%** or modulus, for getting the remainder of the numbers.

1. **Can you apply formula in webpages? Give an example**

Yes, you can. One example would be the conversion table for the following system given in the laboratory exercise number 1 for a different input from the user.

1. **What are the different conditional statements? Describe each**

The different conditional statements are, if statements, executes the codes if the specified condition is true. **If else** statements, executes the code if a condition is true and the other part if it is false. **Elseif** statements are used in executing different codes that have more than two conditions. Lastly, **Switch** statements, for selecting one of many blocks of codes to be executed.

1. **Can you create a condition inside of a condition?**

Yes, you can, it would be called nested conditions. Made up of a condition statement contained within another condition statements.

1. **Do you think these conditional statements is important in the program? Then why?**

The importance of conditional statements in programs are that without them, you would have to do repetitive lines of codes that will affect the speed and efficiency of a program. You won’t also be able to respond to any different options from different scenarios, for example choosing between Option A and Option B.

1. **What are the different looping statements in PHP? Describe each**

The different looping statements are **for** loop, **foreach** loop, **while** loop and **do while** loop. **For** loops through a block of a code in specified number of times, **foreach** loops through a block of code for each element inside an array. **While** loops through a block of code as long as the condition specified is true and lastly, **do while** loops through a block of code once and then repeats depending on the specified condition.

1. **What is the importance of looping?**

The importance of looping is that it repeats a portion of a code in a set number of times until the conditions are met making repetitive tasks shorter and faster as well as minimizing errors and makes your code much more readable.

### REFERENCES

1. <https://www.w3schools.com/css/>
2. <https://www.w3schools.com/html/>
3. <https://www.w3schools.com/php/php_variables.asp>
4. <https://www.w3resource.com/php/operators/arithmetic-operators.php>
5. <https://www.tutorialspoint.com/php/php_arithmatic_operators_examples.htm>
6. [https://www.math10.com/en/algebra/convenrsion-factors-length-area-volume-mass-speed- energy-power-force.html](https://www.math10.com/en/algebra/convenrsion-factors-length-area-volume-mass-speed-energy-power-force.html)
7. <https://www.w3schools.com/php/php_if_else.asp>
8. <https://www.w3schools.com/php/php_switch.asp>
9. <https://www.foxinfotech.in/2019/01/php-form-example-student-grading-system.html>
10. <https://www.w3schools.com/php/php_looping.asp>
11. <https://www.w3schools.com/php/php_looping_while.asp>
12. <https://www.w3schools.com/php/php_looping_do_while.asp>
13. <https://www.w3schools.com/php/php_looping_for.asp>
14. <https://www.w3schools.com/php/php_looping_foreach.asp>
15. <https://www.w3schools.com/php/php_looping_break.asp>

### Note: The following rubrics/metrics will be used to grade students’ output.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Program (100 pts.)** | **(Excellent)** | **(Good)** | **(Fair)** | **(Poor)** |
| **Program execution (20pts)** | Program executes correctly with no syntax or runtime  errors **(18-20pts)** | Program executes with less than 3 errors **(15-17pts)** | Program executes with more than 3 errors **(12-14pts)** | Program does not execute **(10- 11pts)** |
| **Correct output (20pts)** | Program displays correct output with no errors  **(18-20pts)** | Output has minor errors **(15-17pts)** | Output has multiple errors **(12-14pts)** | Output is incorrect  **(10-11pts)** |
| **Design of output (10pts)** | Program displays more than expected **(10pts)** | Program displays minimally expected output  **(8-9pts)** | Program does not display the required output  (**6-7pts)** | Output is poorly designed **(5pts)** |
| **Design of logic (20pts)** | Program is logically well designed **(18- 20pts)** | Program has slight logic errors that do no significantly affect the results  **(15-17pts)** | Program has significant logic errors **(3-5pts)** | Program is incorrect **(10- 11pts)** |
| **Standards (20pts)** | Program code is stylistically well designed **(18- 20pts)** | Few inappropriate design choices (i.e. poor variable names, improper indentation) **(15- 17pts)** | Several inappropriate design choices (i.e. poor variable names, improper indentation) **(12-**  **14pts)** | Program is poorly written **(10-11pts)** |
| **Delivery (10pts)** | The program was delivered on time. **(10pts)** | The program was delivered a day after the deadline. **(8-9pts)** | The program was delivered two days after the deadline. **(6-7pts)** | The program was delivered more than two days after the deadline.  **(5pts)** |