NAMES: **ISHIMWE Gilbert**

Reg number: 21RP00487

PHP Assignment2

1. **PHP** is a general-purpose scripting language geared toward web development.
   It was originally created by **Danish-Canadian** programmer **Rasmus Lerdorf** in 1993 and released in 1995.
   - PHP is an acronym for "PHP: Hypertext Preprocessor" PHP is a widely-used, open source scripting language. PHP scripts are executed on the server. PHP is free to download and use.

2. **Reasons why we use PHP**
   - PHP allows web developers to create dynamic content and interact with databases.
   - **It's easy to learn and use:** One of the main reasons PHP became so commonplace is that it is relatively simple to get started with.
   - **It's versatile**: One of the major benefits of PHP is that it is platform independent, meaning it can be used on Mac OS, Windows, Linux and supports most web browsers.
   - **Open source:** PHP is an open-source project. It is developed and maintained by a worldwide community of developers who make its source code freely available to download and use.
   - **Portability:** PHP runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)

3. The Latest PHP version is: **PHP 8.2**

   List of three Latest version of PHP

   PHP version 8.2:

   PHP 8.2 is the latest PHP version which brings read-only classes, DNF types, null, false, and true types, sensitive parameter redaction support, a new random extension, and several new features along with a few deprecations.

   PHP version 8.1:

PHP 8.1, released in 2021, brings major new features such as Enums, Fibers, never return type, Intersection Types, read-only properties, and more, while ironing out some of its undesired legacy features by deprecating them.

PHP version 8.0:

PHP 8.0, on the 25th year of PHP history, brings several important features such as Union Types, JIT, Constructor Property Promotion, Match Syntax, Named Parameters, and several more performance, syntax, and quality-of-life improvements.

PHP version 7.4

PHP 7.4, the final release in the PHP 7.x series. PHP 7.4 brings typed properties, underscore numeric separator, and other minor improvements to PHP.

4. **different between new release vs stable release of a software product**

A stable release: is a version that has been tested as thoroughly as possible and is as reliable as we can make it. It does not have all the new features of a beta release and it does not have the latest fixes for problems**.**

A new release: is the distribution of the final version or the newest version of a software application. A software release may be public or private and generally signifies the unveiling of a new or upgraded version of the application.

5. **Main features of PHP**
PHP can:

To generate pages and files dynamically.

- To create, open, read, write and close files on the server.
- To collect data from a web form such as user information, email, phone no, etc.
- To send emails to the users of your website (mailing function or Php mailer.
- To send and receive cookies to track the visitor of your website.
- To store, delete, and modify information in your database.
- To restrict unauthorized access to your website.
- To encrypt data for safe transmission over internet. By using many encryption method like:(md5(),sha() and others…).
- PHP 7 supports stricter Type Declarations for function arguments
- PHP 7 supports new operators (like the spaceship operator: <=>)

6. **With a help of examples explain why php is case sensitive?**

In PHP, variable names, function names, and keywords are case-sensitive. This means that a variable named "myCity" is different from a variable named "myCity" or "MyCity". The same applies to function names and keywords.

```
<?php

$myCity = "Kigali";
$MyCity = "Musanze";

echo $myCity; // Outputs "Kigali"
echo $MyCity; // Outputs "Musanze"


?>
<?php

function myFunction() {
  echo "iprc  Tumba";
}

function myfunction() {
  echo "IPRC Kigali";
}

myFunction(); // Outputs "Iprc Tumba"
myfunction(); // Outputs "IPRC Kigali"

?>
```

That's reason why.

7. **What and why do we use comments while writing php codes, With a help of example explain different types of php comments?**

Comments: are used in PHP (and other programming languages) to provide explanations and documentation within the code. They are ignored by the interpreter and do not affect the execution of the program. Comments can be used to explain the purpose of a block of code, provide information about how to use a specific function, or to leave notes for future developers.

There are two types of comments in PHP: single-line and multi-line.

- **Single-line comments**: start with // and continue until the end of the line**. For example**:

// This is a single-line comment.

- **Multi-line comments**: start with /* and end with */. Everything between these symbols is ignored by the interpreter. For example:

  ***/** *this multiline comments*/*

8.

a) **Echo() vs print()**

In PHP, echo and print are used to output text or data to the screen. The main difference between the two is that echo is a language construct, while print is a function. This means that echo does not return a value, while print returns 1 if successful. Additionally, echo can take multiple arguments, while print can only take one.

Here is an example of how echo and print can be used in PHP:
Examples:

```
echo "Hello, World!";
echo "This is a test.";

print("Hello, World!");
print("This is a test.");
```

b) **Print() vs printf()**

In PHP, **print** and **printf** are both used to output text or data to the screen, but they have some key differences.

- ✓ **print** is a function that can only take one argument and returns 1 if successful. It is used to output a single string of text or data to the screen.

  Example:

  ```
  <?php
  print("Hello, World!");
  ?>
  ```

- ✓ **printf** is a function that is used to output a formatted string of text or data to the screen. It takes a format string as its first argument, followed by any number of additional arguments that are used to fill in placeholders within the format string.

  ```
  <?php
  $schoolName = "IPRC Tumba";
  $year = 2;
  printf("My  school name is %s and I am in year  %d IT.", $schoolName, $year);
  ?>
  ```

c) **Printf() vs Print_r():**

The print_r: is used to display human-readable information about a variable.
Example for each:

```
<?php
  $ array = array( "John", "Jacob", "Tom", "Tim");
    print_r($array);
?>
```

Printf(): PHP string printf() function predefined functions. It is used to output a formatted string. We can pass the arg1, arg2, arg++ parameters at percent (%) signs in the main string.

Example:

```
<?php
$age = 67;
$city = "kigali";
printf("There are %u age city is %s.",$number,$str);
?>
```

9. List and Describe different datatype we have in php by categorizing them in scalar, compound and special datatypes.

- ✓ Scalar Types
- ✓ Compound Types
- ✓ Special Types

PHP allows eight different types of data types. All of them are discussed below. The first five are called simple data types and the last three are compound data types.

PHP Data Types: Scalar Types

In simple words, a variable is called scalar type if it holds singular value only. There are 4 scalar data types in PHP.

boolean

integer

float

string

PHP Data Types: Compound Types

In contrast to Scalar data types, a variable is called compound if it holds multiples values within. There are 2 compound data types in PHP.

array

object

PHP Data Types: Special Types

There are 2 special data types in PHP.

resource

NULL

PHP Scalar Data Types

Boolean

Booleans are like a switch which has only two possible values either 1 (true) or 0 (false). Successful events will return true and unsuccessful events return false. NULL type values are also treated as false in Boolean.

```
<!DOCTYPE html>

<head>

   <title> Boolean</title>

</head>

<body>

<?php

// Assign the value TRUE to a variable

$ error = True;

var_dump($error);

?>

</body>

</html>
```

Output:

```
bool(true)
```

Integer

Integers hold only whole numbers including positive and negative numbers, i.e, numbers without fractional part or decimal point. They can be decimal (base 10), octal (base 8) or hexadecimal (base 16). The default base is decimal (base 10).

```
<!DOCTYPE html>

<html lang="en">

<head>

   <title>PHP Integers</title>

</head>

<body>

<?php

$num = 100; // decimal number

var_dump($num);

echo "<br>";

$num = -100; // a negative number

var_dump($num);

echo "<br>";

$hex = 0x1B; // hexadecimal number

var_dump($hex);

echo "<br>";

$oct = 300; // octal number

var_dump($oct);

?>

</body>

</html>
```

PHP Floating Point Numbers or Doubles

Floating point numbers (also known as “floats”, “doubles”, or “real numbers”) are decimal or fractional numbers like demonstrated in the example below.

Example:

```
<!DOCTYPE html>
```

```
<html lang="en">
<head>
   <title>PHP Floats</title>
</head>
<body>
<?php
$num = 1.8;
var_dump($num);
echo "<br>";
$num2 = 10.2e3;
var_dump($num2);
echo "<br>";
$num3 = 4E-12;
var_dump($num3);
?>
</body>
</html>
```

String

In a string, letters or any alphabets, even numbers are included. These are written within double quotes during declaration. The strings can also be written within single quotes but it will be treated differently while printing variables. To clarify this look at the example below.

```
<!DOCTYPE html>
<html lang="en">
<head>
   <title>PHP Strings</title>
</head>
<body>
<?php
```

```
$a = 'Hello world!';
echo $a;
echo "<br>";
$b = "Hello world!";
echo $b;
echo "<br>";
$c = 'Stay here, I\'ll be back.';
echo $c;
?>
</body>
</html>
```

Output:

```
Hello world!
Hello world!
Stay here, I'll be back.
```

PHP Compound DataTypes

Array

An array is a variable that can hold more than one value at a time. It is useful to aggregate a series of related items together, for example, a set of country or city names.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <title>PHP Arrays</title>
</head>
<body>
<?php
$colors = array("Red", "Green", "Blue");
var_dump($colors);
```

```
echo "<br>";
$color_codes = array(
    "Red" => "#ff0000",
    "Green" => "#00ff00",
    "Blue" => "#0000ff"
);
var_dump($color_codes);
?>
</body>
</html>
```

Output:

```
array(3) { [0]=> string(3) "Red" [1]=> string(5) "Green" [2]=> string(4) "Blue" }
array(3) { ["Red"]=> string(7) "#ff0000" ["Green"]=> string(7) "#00ff00" ["Blue"]=>
string(7) "#0000ff" }
```

Object

An object is a data type which stores not only data but also information on how to process that data. Unlike the other data types in PHP, an object must be explicitly declared.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>PHP Objects</title>
</head>
<body>
<?php
// Class definition
class greeting{
    // properties
    public $str = "Hello World!";
    // methods
    function show_greeting(){
        return $this->str;
    }
}
// Create object from class
$message = new greeting;
var_dump($message);
?>
</body>
</html>
```

Output:

```
object(greeting)#1 (1) { ["str"]=> string(12) "Hello World!" }
```

PHP Special Data Types

Null

The special NULL value is used to represent empty variables in PHP. A variable of type NULL is a variable without any data. NULL is the only possible value of type null. Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>PHP NULL Value</title>
</head>
<body>
<?php
$a = NULL;
var_dump($a);
echo "<br>";
```

```
$b = "Hello World!";
$b = NULL;
var_dump($b);
?>
</body>
</html>
```

Output:

```
NULL
 NULL
```

Resources

A resource is a special variable, holding a reference to an external resource.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>PHP Resources</title>
</head>
<body>
<?php
// Open a file for reading
$handle = fopen("note.txt", "r");
var_dump($handle);
?>
</body>
</html>
```

Output:

```
resource(2) of type (stream)
```

10.

- PHP variables are characters that stores value or information such as text or integers in your code. It is important to know that variables in PHP are usually represented by a dollar sign($) followed by the name of the variable.
- Rules for PHP variables:
  - ✓ A variable starts with the $ sign, followed by the name of the variable
  - ✓ A variable name must start with a letter or the underscore character
  - ✓ A variable name cannot start with a number
  - ✓ A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
  - ✓ Variable names are case-sensitive ($age and $AGE are two different variables)

11. explanations for at least 10 super global variables?

Below is the list of super global variables available in PHP:

- ❖ $GLOBALS
- ❖ $_SERVER
- ❖ $_REQUEST
- ❖ $_GET
- ❖ $_POST
- ❖ $_SESSION
- ❖ $_COOKIE
- ❖ $_FILES
- ❖ $_ENV
- ➢ $GLOBALS: It is a super global variable which is used to access global variables from anywhere in the PHP script. PHP stores all the global variables in array $GLOBALS [] where index holds the global variable name, which can be accessed.
  Eg:
  ```
  <?php
  echo $_SERVER['PHP_SELF'];
  echo "<br>";
  echo $_SERVER['SERVER_NAME'];
  echo "<br>";
  echo $_SERVER['HTTP_HOST'];
  echo "<br>";
  echo $_SERVER['HTTP_USER_AGENT'];
  echo "<br>";
  echo $_SERVER['SCRIPT_NAME'];
  echo "<br>"
  ?>
  ```
- ➢ $_REQUEST : It is a superglobal variable which is used to collect the data after submitting a HTML form. $_REQUEST is not used mostly, because $_POST and $_GET perform the same task and are widely used.

EG.

```
<!DOCTYPE html>
<html>
<body>
<form method="post" action="<?php echo $_SERVER['PHP_SELF'];?>">
 NAME: <input type="text" name="sex">
 <button type="submit">SUBMIT</button>
</form>
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $name = htmlspecialchars($_REQUEST['sex']);
    if(empty($sex)){
        echo "sex is empty";
    } else {
        echo $sex;
    }
}
?>
</body>
</html>
```

- $_POST : It is a super global variable used to collect data from the HTML form after submitting it. When form uses method post to transfer data, the data is not visible in the query string, because of which security levels are maintained in this method.

Eg.

```
<!DOCTYPE html>
<html>
<body>

<form method="post" action="<?php echo $_SERVER['PHP_SELF'];?>">
 <label for="name">Please enter your name: </label>
 <input name="name" type="text"><br>
 <label for="age">Please enter your age: </label>
 <input name="age" type="text"><br>
 <input type="submit" value="Submit">
 <button type="submit">SUBMIT</button>
</form>
<?php
$nm=$_POST['name'];
$age=$_POST['age'];
echo "<strong>".$nm." is $age years old.</strong>";
?>
</body>
</html>
```

- $_GET : $_GET is a super global variable used to collect data from the HTML form after submitting it. When form uses method get to transfer data, the data is visible in the query string, therefore the values are not hidden. $_GET super global array variable stores the values that come in the URL.

  Eg .

  ```
  <!DOCTYPE html>
  <html>
  <head>
  <title></title>
  </head>
  <body bgcolor="cyan">
      <?php
          $name = $_GET['name'];
          $city = $_GET['city'];
          echo "<h1>This is ".$name." of ".$city."</h1><br>";
      ?>
      <img src = "2.jpg" alt = "nanilake" height = "400" width="500" />
  </body>
  </html>
  ```

References

Ayesh. (n.d.). *https://php.watch/versions#:~:text=PHP%208.1%2C%20released%20in%202021,legacy%20features%20by%20deprecating%20them.* Retrieved from ww.php.watch.com.

Careerera. (n.d.). *https://www.careerera.com/blog/features-of-php-programming-language*. Retrieved from www.careerera.com.

company, r. (n.d.). *https://codeberryschool.com/blog/en/beginners-guide-to-php-programming/*. Retrieved from www.codeberryschool.com.

Data, R. (n.d.). *https://www.w3schools.com/php/php_variables.asp*. Retrieved from www.w3schools.com.

geeksforgeeks. (n.d.). *https://www.geeksforgeeks.org/php-superglobals/*. Retrieved from www.geeksforgeeks.org.

Ltd, X. (n.d.). *https://forums.freebsd.org/threads/stable-vs-release-difference.48388/#:~:text=Normal%20users%20choose%20the%20latest,and%203rd%2Dparty%20applications).* Retrieved from www.forums.freebsd.org.

phptutorial.net. (n.d.). *https://www.phptutorial.net/php-tutorial/php-comments/*. Retrieved from www.phptutorial.net.

Sebhastian, N. (n.d.). *https://sebhastian.com/is-php-case-sensitive/#:~:text=PHP%20classes%20are%20a%20mix,they%20are%20partially%20case%2Dsensitive.&text=As%20you%20can%20see%20in,error%3A%20cannot%20redeclare%20the%20function.* Retrieved from www.sebhastian.com.

trytoprogram. (n.d.). *https://www.trytoprogram.com/php/php-data-types/#data*. Retrieved from www.trytoprogram.com.