

NEW EDITION

PHP

**UNIVERSITY ­ OF MUMBAI**

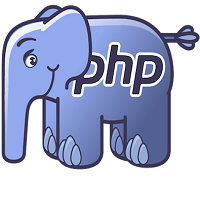
BY:

PRIYUSH B.K.

2022

COPYRIGHT © 2022 ALL RIGHTS RESERVES

**📍What is PHP**

PHP is an **open-source, interpreted, and object-oriented scripting language t**hat can be executed at the server-side.

PHP was created by **Rasmus Lerdorf in 1994** but appeared in the market in 1995. PHP 7.4.0 is the latest version of **PHP**, which was released on 28 November.

* PHP is well suited for web development.
* Therefore, it is used to develop web applications (an application that executes on the server and generates the dynamic page.).
* 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.
* PHP is faster than other scripting languages, for example, ASP and JSP.
* PHP is a server-side scripting language, which is used to manage the dynamic content of the website.
* PHP can be embedded into HTML.
* PHP is an object-oriented language.
* PHP is an open-source scripting language.
* PHP is simple and easy to learn language.

**🖋Why use PHP:**

* PHP is a server-side scripting language, which is used to design the dynamic web applications with MySQL database.
* It handles dynamic content, database as well as session tracking for the website.
* PHP runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
* PHP is compatible with almost all servers used today (Apache, IIS, etc.)
* PHP supports a wide range of databases
* PHP is free. Download it from the official PHP resource: www.php.net
* PHP is easy to learn and runs efficiently on the server side
* You can create sessions in PHP.
* It can access cookies variable and also set cookies.
* It helps to encrypt the data and apply validation.
* PHP supports several protocols such as HTTP, POP3, SNMP, LDAP, IMAP, and many more.
* Using PHP language, you can control the user to access some pages of your website.
* As PHP is easy to install and set up, this is the main reason why PHP is the best language to learn

**🖊What is a PHP File?**

* PHP files can contain text, HTML, CSS, JavaScript, and PHP code
* PHP code is executed on the server, and the result is returned to the browser as plain HTML
* PHP files have extension ".php"

**🔗PHP Features:**

**✔Performance:**

PHP script is executed much faster than those scripts which are written in other languages such as JSP and ASP. PHP uses its own memory, so the server workload and loading time is automatically reduced, which results in faster processing speed and better performance.

**✔Open Source:**

PHP source code and software are freely available on the web All its components are free to download and use.

**✔Familiarity with syntax:**

PHP has easily understandable syntax. Programmers are comfortable coding with it.

**✔Embedded**:

PHP code can be easily embedded within HTML tags and script.

**✔Platform Independent:**

PHP is available for WINDOWS, MAC, LINUX & UNIX operating system. A PHP application developed in one OS can be easily executed in other OS also.

**✔Database Support:**

PHP supports all the leading databases such as MySQL, SQLite, ODBC, etc.

**✔Error Reporting -**

PHP has predefined error reporting constants to generate an error notice or warning at runtime. E.g., E\_ERROR, E\_WARNING, E\_STRICT, E\_PARSE.

**✔Loosely Typed Language:**

PHP allows us to use a variable without declaring its datatype. It will be taken automatically at the time of execution based on the type of data it contains on its value.

**✔Web servers Support:**

PHP is compatible with almost all local servers used today like Apache, Netscape, Microsoft IIS, etc.

**✔Security**:

PHP is a secure language to develop the website. It consists of multiple layers of security to prevent threads and malicious attacks.

**✔Control**:

Different programming languages require long script or code, whereas PHP can do the same work in a few lines of code. It has maximum control over the websites like you can make changes easily whenever you want.

**✔A Helpful PHP Community:**

It has a large community of developers who regularly updates documentation, tutorials, online help, and FAQs. Learning PHP from the communities is one of the significant benefits.

**🌐Install PHP**

To install PHP, we will suggest you to install AMP (Apache, MySQL, PHP) software stack. It is available for all operating systems. There are many AMP options available in the market that are given below:

* WAMP for Windows
* LAMP for Linux
* MAMP for Mac
* SAMP for Solaris
* FAMP for FreeBSD
* XAMPP

**PHP Syntax PHP Ex**

<!DOCTYPE html>  
<html>  
<body>  
<h1>My first PHP page</h1>  
<?php  
echo "Hello World!";  
?>  
</body>  
</html>

Output: My first PHP page

Hello World!

1. **<?php**
2. //your code here
3. **?>**

**📎PHP Comments**

* PHP comments can be used to describe any line of code so that other developer can understand the code easily. It can also be used to hide any code.
* PHP supports single line and multi-line comments. These comments are similar to C/C++ and Perl style (Unix shell style) comments.

**📎PHP Single Line Comments**

* There are two ways to use single line comments in PHP.
* // (C++ style single line comment)
* # (Unix Shell style single line comment)

1. <?php
2. // this is C++ style single line comment
3. # this is Unix Shell style single line comment
4. echo "Welcome to PHP single line comments";
5. ?>

**📎PHP Multi Line Comments**

In PHP, we can comment multiple lines also. To do so, we need to enclose all lines within /\* \*/. Let's see a simple example of PHP multiple line comment

1. <?php
2. /\* Anything placed within comment will not be displayed on the browser; \*/
3. echo "Welcome to PHP multi line comment";
4. ?>

**📎PHP echo and print Statements**

echo and print are more or less the same. They are both used to output data to the screen.

The differences are small: echo has no return value while print has a return value of 1 so it can be used in expressions. echo can take multiple parameters (although such usage is rare) while print can take one argument. echo is marginally faster than print.

**The PHP echo Statement** The echo statement can be used with or without parentheses: echo or echo().

**The PHP print Statement** The print statement can be used with or without parentheses: print or print().

**📕PHP Variables:**

In PHP, a variable is declared using a $ sign followed by the variable name. Here, some important points to know about variables:

* As PHP is a loosely typed language, so we do not need to declare the data types of the variables. It automatically analyzes the values and makes conversions to its correct datatype.
* After declaring a variable, it can be reused throughout the code.
* Assignment Operator (=) is used to assign the value to a variable.

Syntax of declaring a variable in PHP is given below:

**$variablename=value;**

***Rules for declaring PHP variable:***

* A variable must start with a dollar ($) sign, followed by the variable name.
* It can only contain alpha-numeric character and underscore (A-z, 0-9, \_).
* A variable name must start with a letter or underscore (\_) character.
* A PHP variable name cannot contain spaces.
* One thing to be kept in mind that the variable name cannot start with a number or special symbols.
* PHP variables are case-sensitive, so $name and $NAME both are treated as different variable.

**OUTPUT:**

**string is: hello string**

**integer is: 200**

**float is: 44.6**

1. **<?php**
2. $str="hello string";
3. $x=200;
4. $y=44.6;
5. echo "string is: $str **<br/>**";
6. echo "integer is: $x **<br/>**";
7. echo "float is: $y **<br/>**";
8. **?>**

**OUTPUT:**

**11**

1. **<?php**
2. $x=5;
3. $y=6;
4. $z=$x+$y;
5. echo $z;
6. **?>**

**📕PHP Variable Scope**

The scope of a variable is defined as its range in the program under which it can be accessed. In other words, "The scope of a variable is the portion of the program within which it is defined and can be accessed."

PHP has three types of variable scopes:

* Local variable
* Global variable
* Static variable

**🚩PHP Data Types**

PHP data types are used to hold different types of data or values. PHP supports 8 primitive data types that can be categorized further in 3 types:

* Scalar Types (predefined)
* Compound Types (user-defined)
* Special Types

**PHP Data Types: Scalar Types**

It holds only single value. There are 4 scalar data types in PHP.

* boolean
* integer
* float
* string

**PHP Data Types: Compound Types**

It can hold multiple values. 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 Boolean**

Booleans are the simplest data type works like switch. It holds only two values: TRUE (1) or FALSE (0). It is often used with conditional statements. If the condition is correct, it returns TRUE otherwise FALSE.

**Example:**

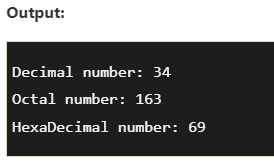
1. <?php
2. **if** (TRUE)
3. echo "This condition is TRUE.";
4. **if** (FALSE)
5. echo "This condition is FALSE.";
6. ?>

o/t: This condition is TRUE.

**📍PHP Integer**

Integer means numeric data with a negative or positive sign. It holds only whole numbers, i.e., numbers without fractional part or decimal points.

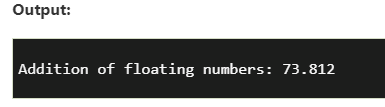
Rules for integer:

* An integer can be either positive or negative.
* An integer must not contain decimal point.
* Integer can be decimal (base 10), octal (base 8), or hexadecimal (base 16).
* The range of an integer must be lie between 2,147,483,648 and 2,147,483,647 i.e., -2^31 to 2^31.

1. <?php
2. $dec1 = 34;
3. $oct1 = 0243;
4. $hexa1 = 0x45;
5. echo "Decimal number: " .$dec1. "</br>";
6. echo "Octal number: " .$oct1. "</br>";
7. echo "HexaDecimal number: " .$hexa1. "</br>";
8. ?>

**📍PHP Float**

A floating-point number is a number with a decimal point. Unlike integer, it can hold numbers with a fractional or decimal point, including a negative or positive sign.

**Example:**

1. <?php
2. $n1 = 19.34;
3. $n2 = 54.472;
4. $sum = $n1 + $n2;
5. echo "Addition of floating numbers: " .$sum;
6. ?>

**📍PHP String**

* A string is a non-numeric data type. It holds letters or any alphabets, numbers, and even special characters.
* String values must be enclosed either within single quotes or in double quotes.

**Example:**

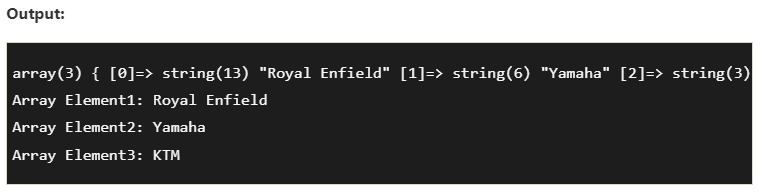
1. <?php
2. $company = "Javatpoint";
3. //both single and double quote statements will treat different
4. echo "Hello $company";
5. echo "</br>";
6. echo 'Hello $company';
7. ?>



**📍PHP Array**

An array is a compound data type. It can store multiple values of same data type in a single variable.

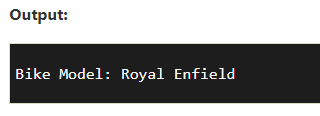
1. <?php
2. $bikes = **array** ("Royal Enfield", "Yamaha", "KTM");
3. var\_dump($bikes);   //the var\_dump() function returns the datatype and values
4. echo "</br>";
5. echo "Array Element1: $bikes[0] </br>";
6. echo "Array Element2: $bikes[1] </br>";
7. echo "Array Element3: $bikes[2] </br>";
8. ?>

**📍****PHP object**

Objects are the instances of user-defined classes that can store both values and functions. They must be explicitly declared.

**Example:**

1. <?php
2. **class** bike {
3. **function** model() {
4. $model\_name = "Royal Enfield";
5. echo "Bike Model: " .$model\_name;
6. }
7. }
8. $obj = **new** bike();
9. $obj -> model();
10. ?>



**📍PHP Resource**

Resources are not the exact data type in PHP. Basically, these are used to store some function calls or references to external PHP resources. For example - a database call. It is an external resource.

**📍PHP Null**

Null is a special data type that has only one value: NULL. There is a convention of writing it in capital letters as it is case sensitive.

**Example:**

1. <?php
2. $nl = NULL;
3. echo $nl;   //it will not give any output
4. ?>

**✂PHP Functions**

PHP function is a piece of code that can be reused many times. It can take input as argument list and return value. There are thousands of built-in functions in PHP.

In PHP, we can define Conditional function, Function within Function and Recursive function also.

**🚩Advantage of PHP Functions**

* **Code Reusability**: PHP functions are defined only once and can be invoked many times, like in other programming languages.
* **Less Code**: It saves a lot of code because you don't need to write the logic many times. By the use of function, you can write the logic only once and reuse it.
* **Easy to understand:** PHP functions separate the programming logic. So it is easier to understand the flow of the application because every logic is divided in the form of functions.

**📍PHP User-defined Functions:**

We can declare and call user-defined functions easily. Let's see the syntax to declare user-defined functions.

Syntax

*File: function1.php*

1. <?php
2. **function** sayHello(){
3. echo "Hello PHP Function";
4. }
5. sayHello();//calling function
6. ?>
7. **function** functionname(){
8. //code to be executed
9. }

Output:

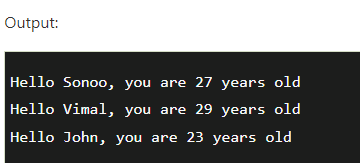
Hello PHP Function

**📍PHP Function Arguments:**

* We can pass the information in PHP function through arguments which is separated by comma.
* PHP supports Call by Value (default), Call by Reference, Default argument values and Variable-length argument list.

*File: functionarg.php*

1. <?php
2. **function** sayHello($name,$age){
3. echo "Hello $name, you are $age years old<br/>";
4. }
5. sayHello("Sonoo",27);
6. sayHello("Vimal",29);
7. sayHello("John",23);
8. ?>



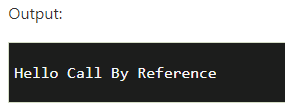
**📍PHP Call By Reference:**

* Value passed to the function doesn't modify the actual value by default (call by value). But we can do so by passing value as a reference.
* By default, value passed to the function is call by value. To pass value as a reference, you need to use ampersand (&) symbol before the argument name.

Ex .

*File: functionref.php*

1. <?php
2. **function** adder(&$str2)
3. {
4. $str2 .= 'Call By Reference';
5. }
6. $str = 'Hello ';
7. adder($str);
8. echo $str;
9. ?>

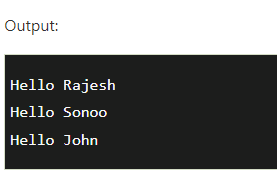


**📍PHP Function: Default Argument Value**

We can specify a default argument value in function. While calling PHP function if you don't specify any argument, it will take the default argument. Let's see a simple example of using default argument value in PHP function.

*File: functiondefaultarg.php*

1. <?php
2. **function** sayHello($name="Sonoo"){
3. echo "Hello $name<br/>";
4. }
5. sayHello("Rajesh");
6. sayHello();//passing no value
7. sayHello("John");
8. ?>



**📍PHP Function: Returning Value:**

Let's see an example of PHP function that returns value.

Ex.

*File: functiondefaultarg.php*

1. <?php
2. **function** cube($n){
3. **return** $n\*$n\*$n;
4. }
5. echo "Cube of 3 is: ".cube(3);
6. ?>



[**https://github.com/Prime-Inspire/CodeIgniter-EMS**](https://github.com/Prime-Inspire/CodeIgniter-EMS)

[**https://github.com/Prime-Inspire/php-mysql-ems**](https://github.com/Prime-Inspire/php-mysql-ems)

**Chapter 5-PHP**

# **5.7 PHP Variables**

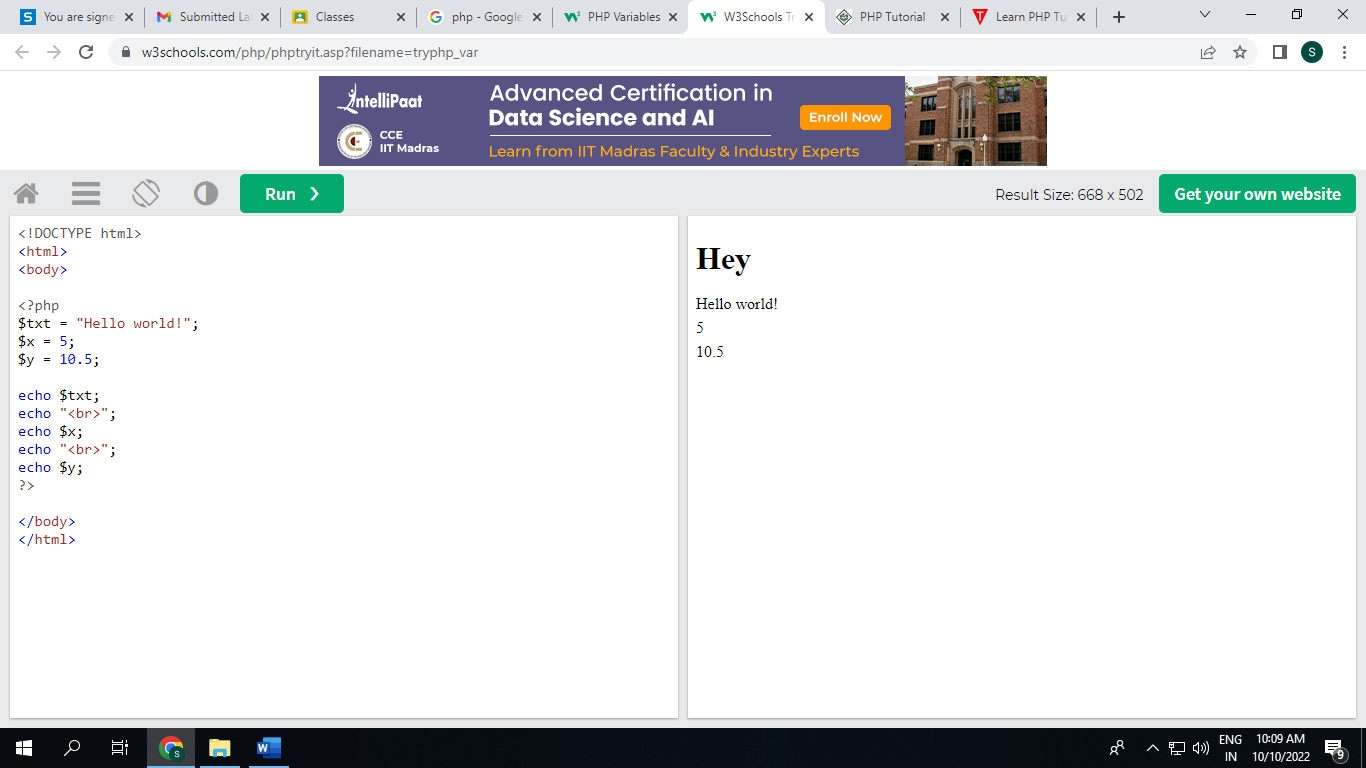
* Variables are "containers" for storing information.
* They are case sensitive

## 1.Creating (Declaring) PHP Variables

In PHP, a variable starts with the $ sign, followed by the name of the variable:

### Example

<?php  
$txt = "Hello world!";  
$x = 5;  
$y = 10.5;  
?>



**5.7.1 PHP Variables**

A variable can have a short name (like x and y) or a more descriptive name (age, carname, total\_volume).

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)

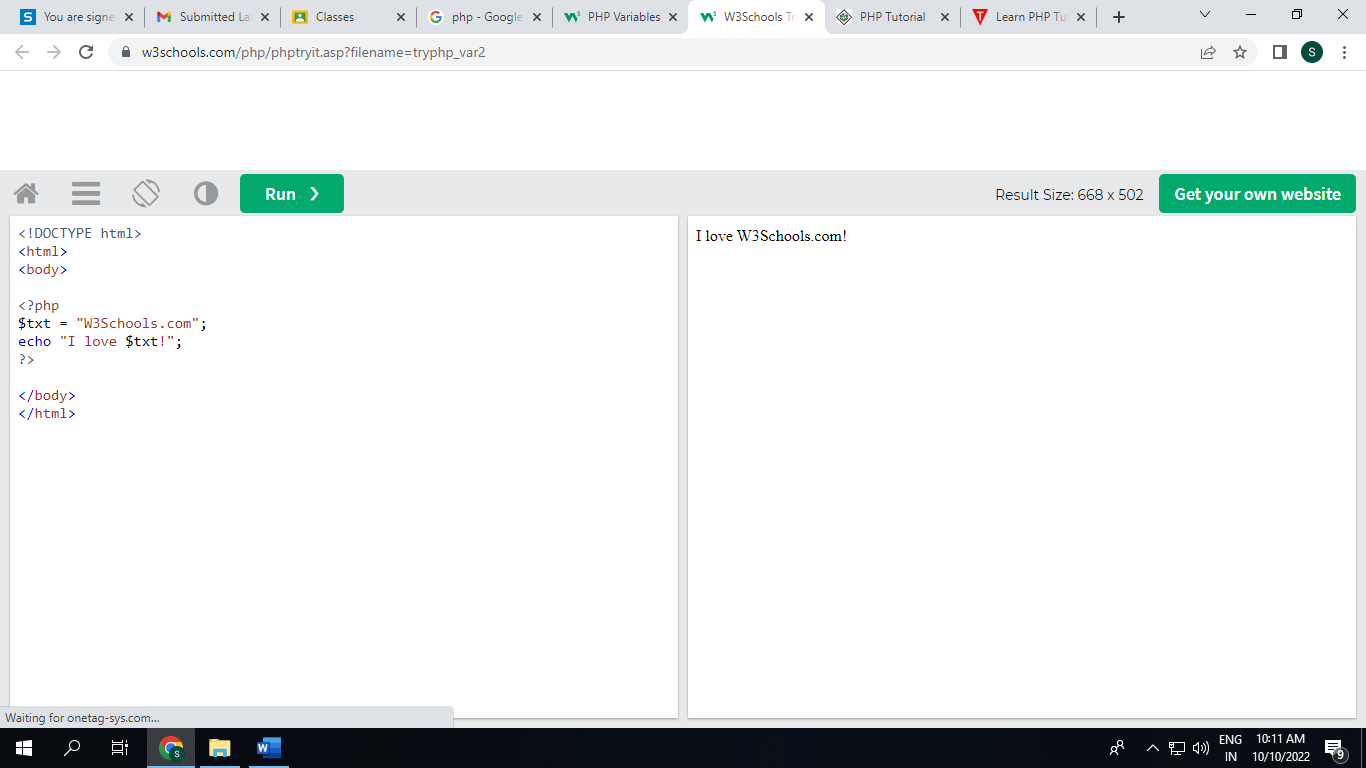
## 5.7.2 Output Variables

The PHP echo statement is often used to output data to the screen.

The following example will show how to output text and a variable:

### Example 1

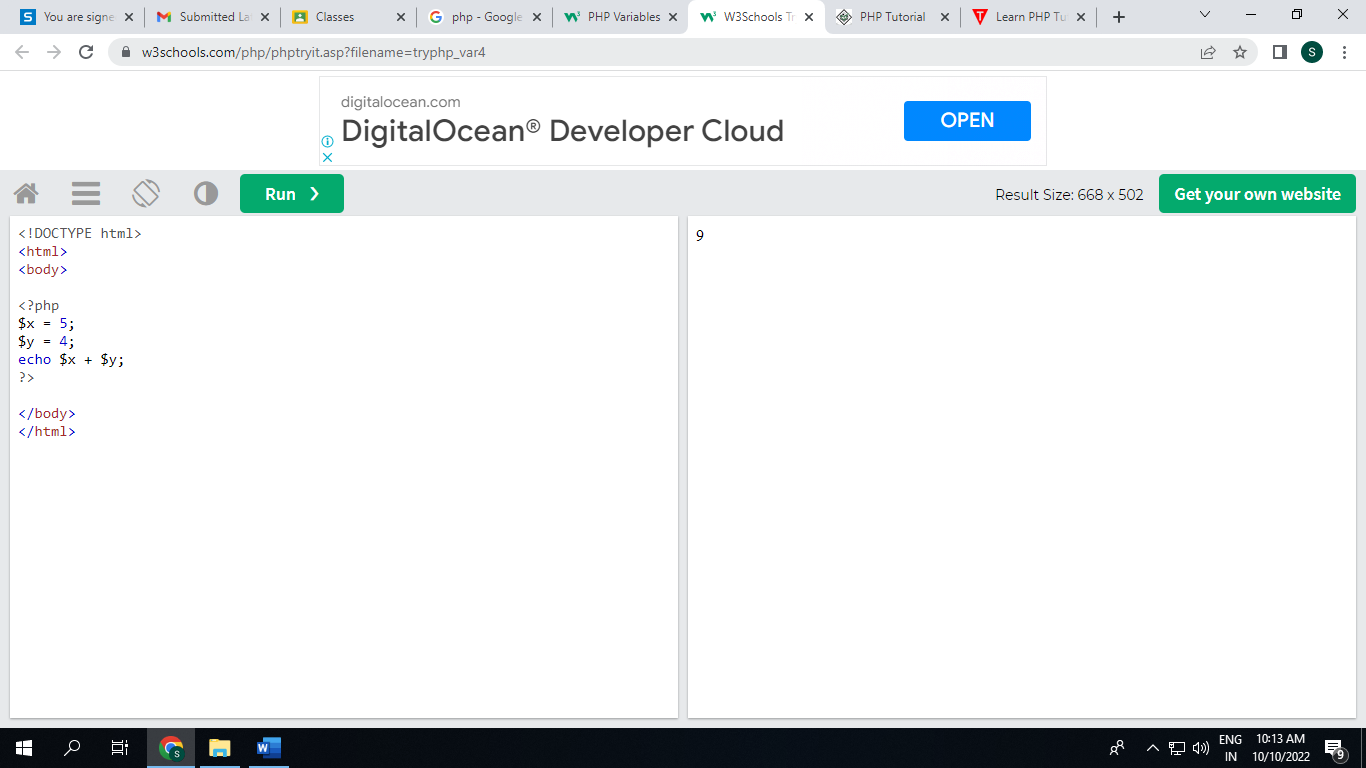
<?php  
$txt = "W3Schools.com";  
echo "I love $txt!";  
?>



### Example 2

sum of two variables:

<?php  
$x = 5;  
$y = 4;  
echo $x + $y;  
?>



# **5.8 PHP Data Types**

Variables can store data of different types, and different data types can do different things.

PHP supports the following data types:

* String
* Integer
* Float (floating point numbers - also called double)
* Boolean
* Array
* Object
* NULL
* Resource

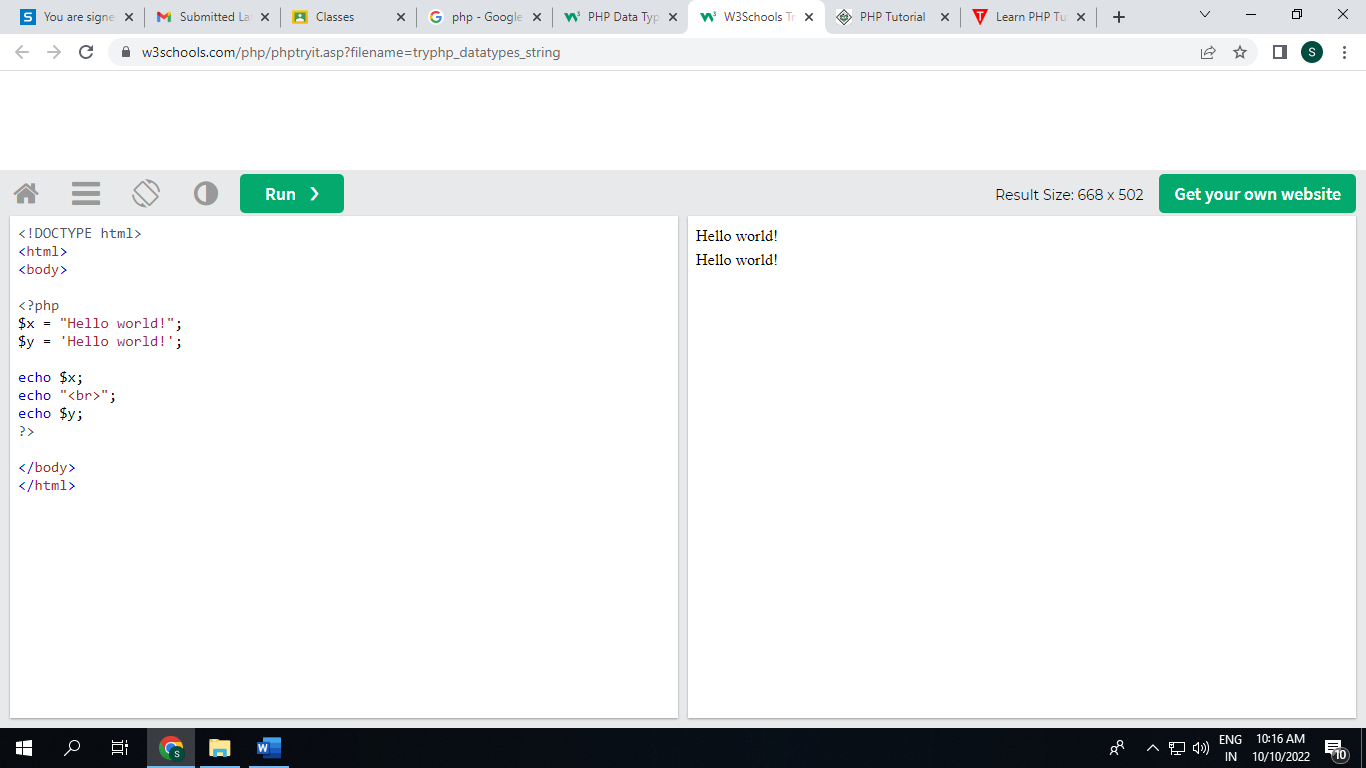
## 1.PHP String

A string is a sequence of characters, like "Hello world!".

A string can be any text inside quotes. You can use single or double quotes:

### Example

<?php  
$x = "Hello world!";  
$y = 'Hello world!';  
  
echo $x;  
echo "<br>";  
echo $y;  
?>



**2. PHP Integer**

An integer data type is a non-decimal number between -2,147,483,648 and 2,147,483,647.

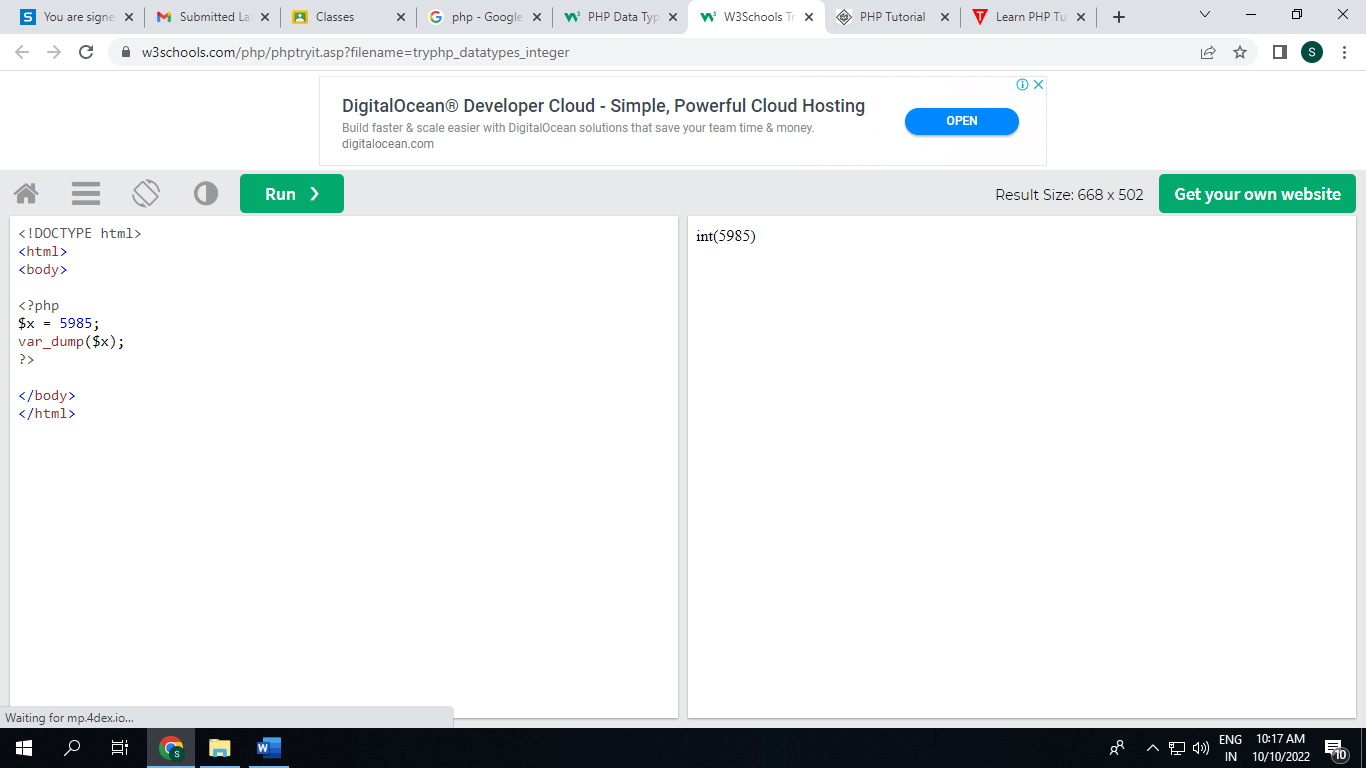
Rules for integers:

* An integer must have at least one digit
* An integer must not have a decimal point
* An integer can be either positive or negative
* Integers can be specified in: decimal (base 10), hexadecimal (base 16), octal (base 8), or binary (base 2) notation

In the following example $x is an integer. The PHP var\_dump() function returns the data type and value:

### Example

### <?php $x = 5985; var\_dump($x); ?>



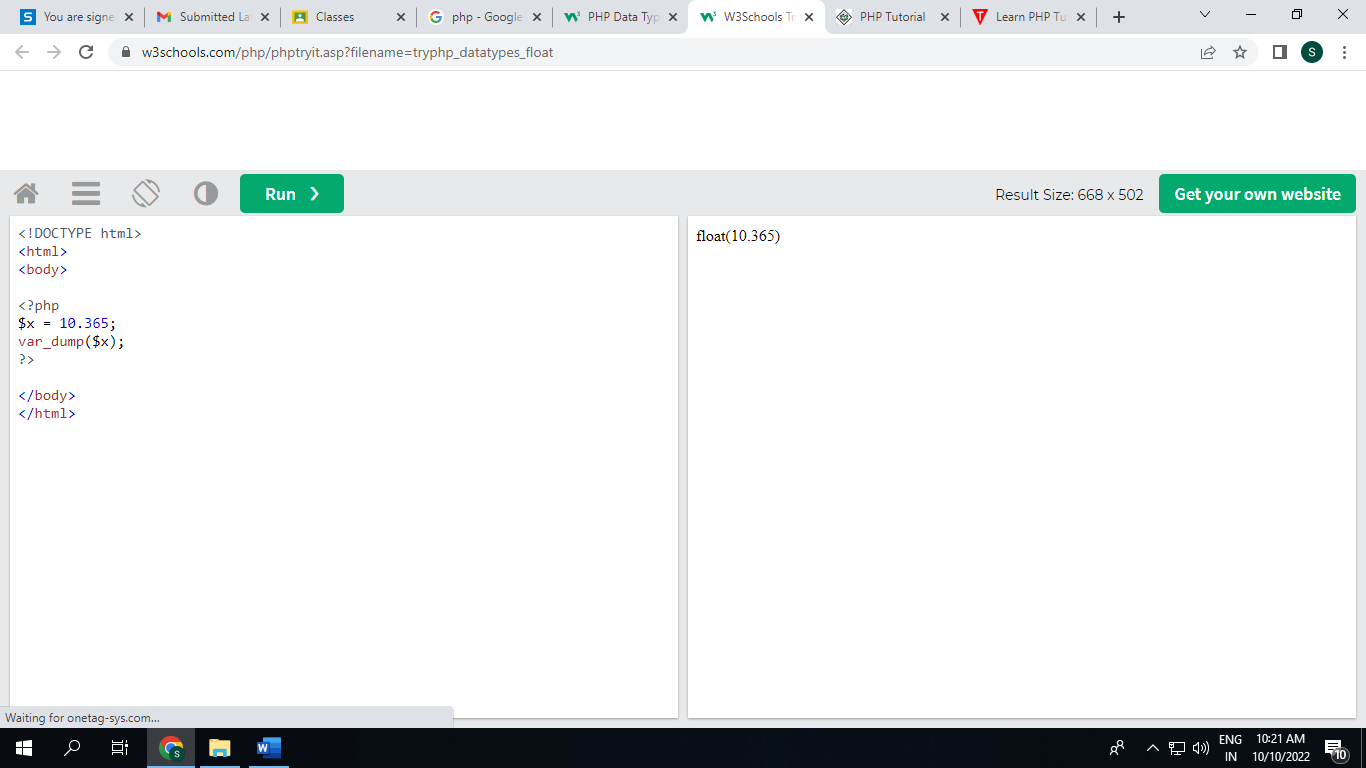
## 3.PHP Float

A float (floating point number) is a number with a decimal point or a number in exponential form.

In the following example $x is a float. The PHP var\_dump() function returns the data type and value:

### Example

<?php  
$x = 10.365;  
var\_dump($x);  
?>



**4.PHP Boolean**

A Boolean represents two possible states: TRUE or FALSE.

$x = true;  
$y = false;

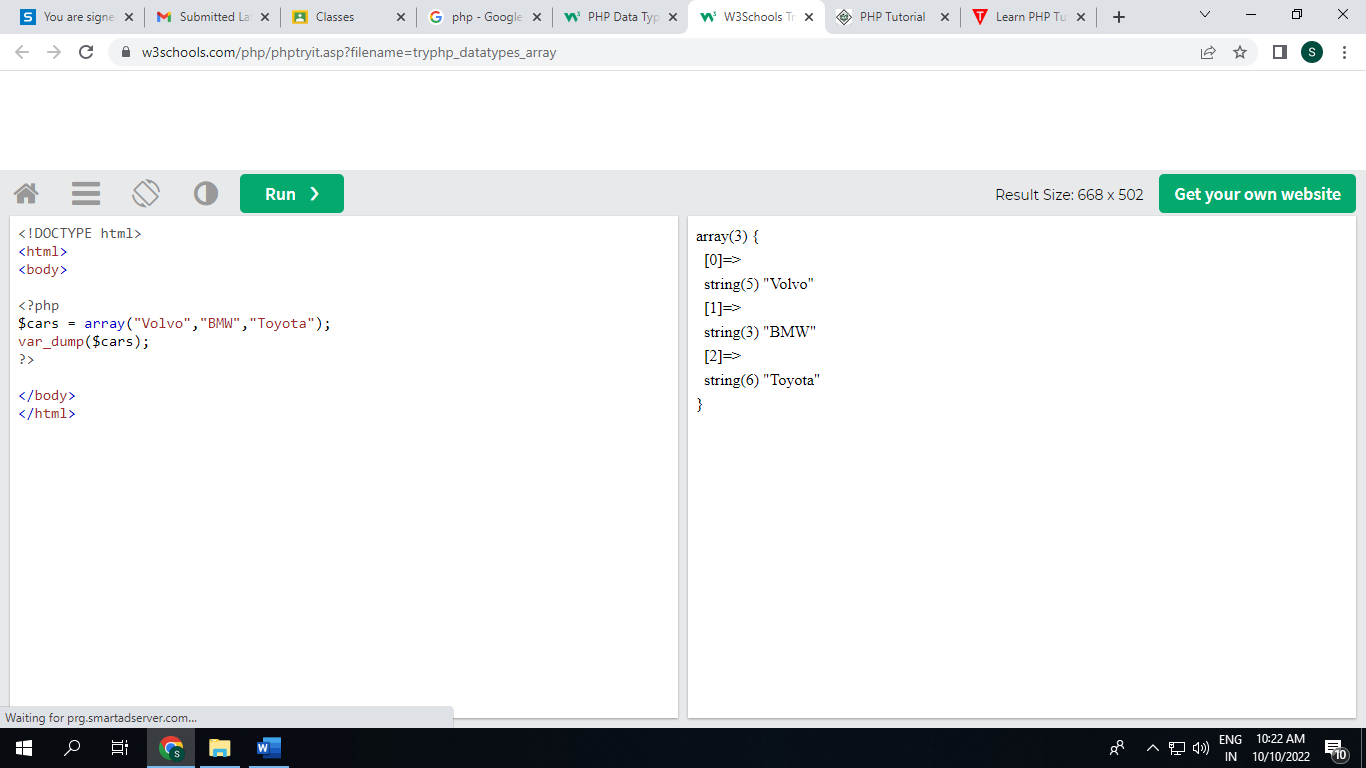
## 5.PHP Array

An array stores multiple values in one single variable.

In the following example $cars is an array. The PHP var\_dump() function returns the data type and value:

### Example

<?php  
$cars = array("Volvo","BMW","Toyota");  
var\_dump($cars);  
?>



## 5.9 PHP Built-in Functions

PHP has over 1000 built-in functions that can be called directly, from within a script, to perform a specific task.

**PHP User Defined Functions**

* A function is a block of statements that can be used repeatedly in a program.
* A function will not execute automatically when a page loads.
* A function will be executed by a call to the function.

## Create a User Defined Function in PHP

## A user-defined function declaration starts with the word function:

### **Syntax**

## function *functionName*() { *code to be executed*; }

## **Note:** A function name must start with a letter or an underscore. Function names are NOT case-sensitive.

### Example

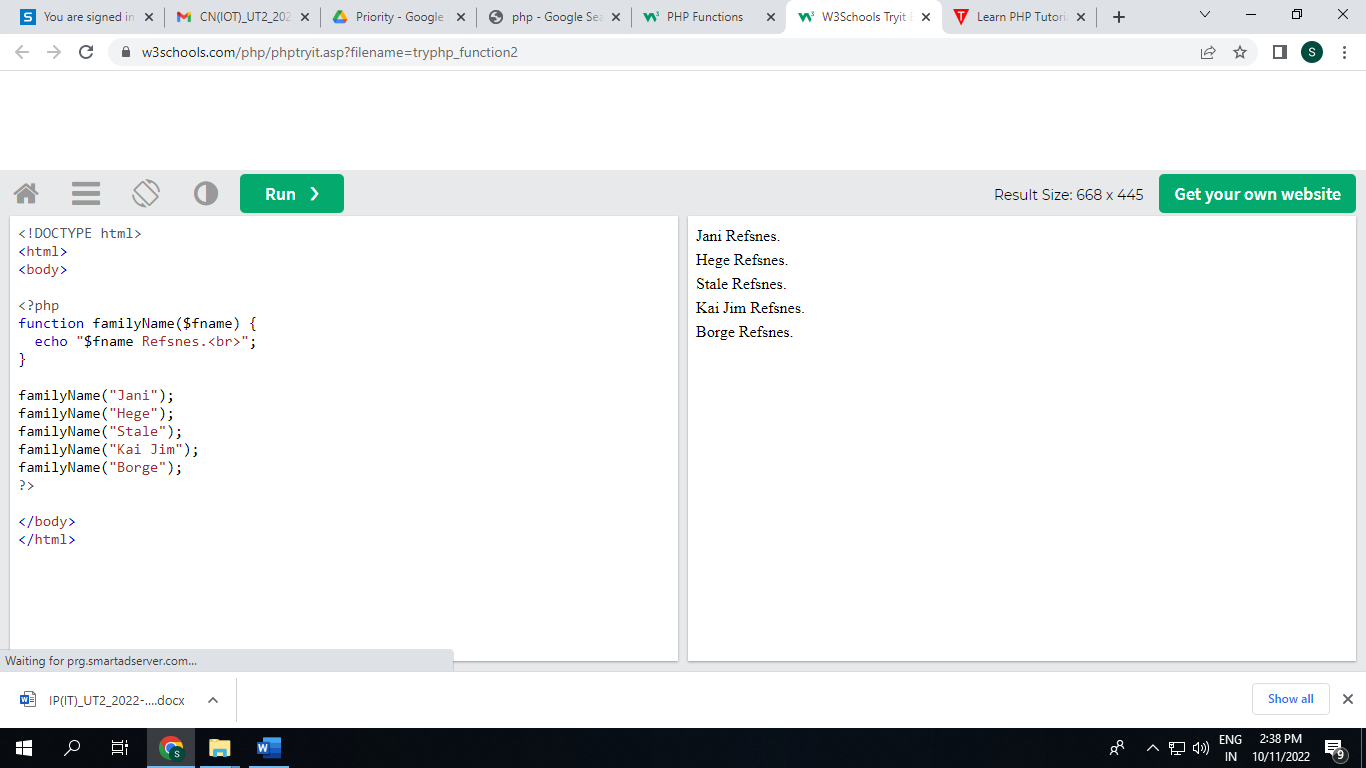
## <?php function writeMsg() {   echo "Hello world!"; } writeMsg(); // call the function ?>

## 5.10 PHP Function Arguments

## Arguments are specified after the function name, inside the parentheses. You can add as many arguments as you want, just separate them with a comma.

### Example

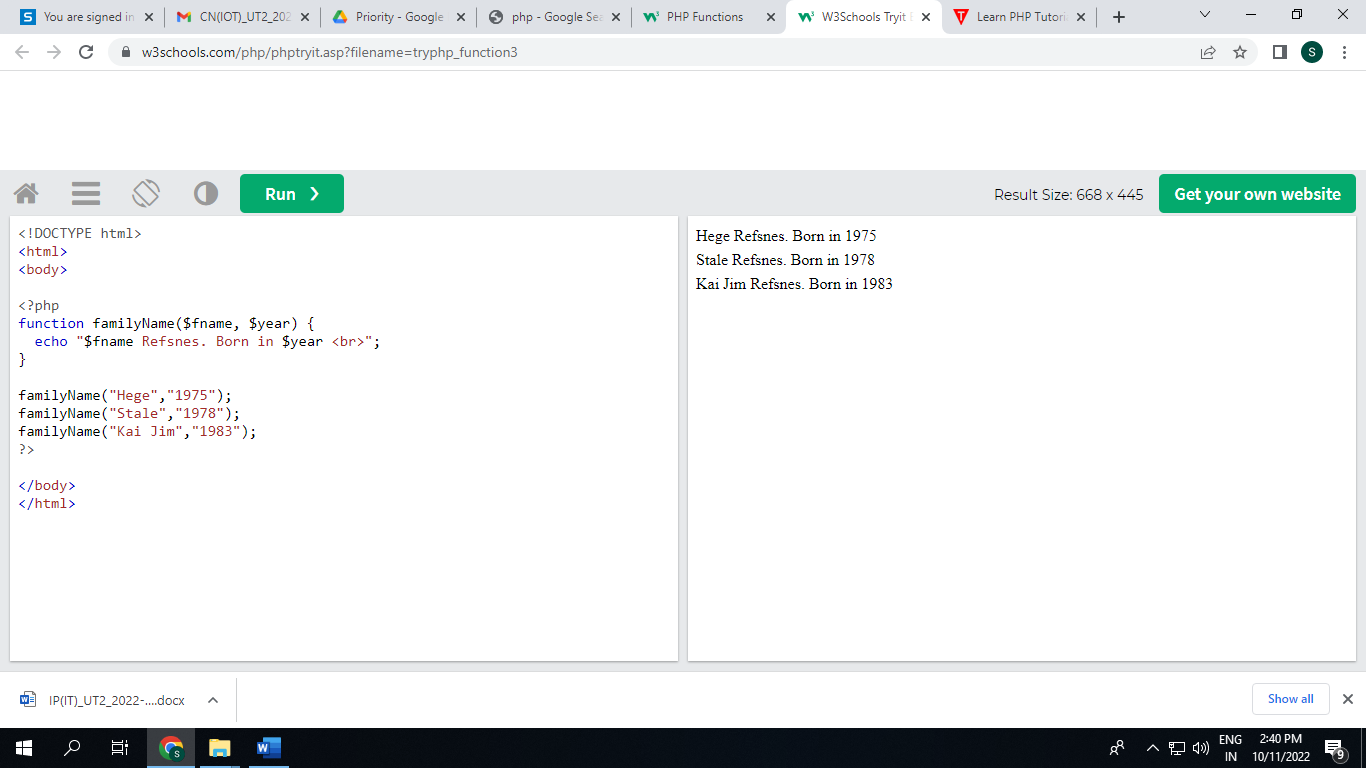
<?php  
function familyName($fname) {  
  echo "$fname Refsnes.<br>";  
}  
  
familyName("Jani");  
familyName("Hege");  
familyName("Stale");  
familyName("Kai Jim");  
familyName("Borge");  
?>



The following example has a function with two arguments ($fname and $year):

### Example

<?php  
function familyName($fname, $year) {  
  echo "$fname Refsnes. Born in $year <br>";  
}  
  
familyName("Hege", "1975");  
familyName("Stale", "1978");  
familyName("Kai Jim", "1983");  
?>



## 5.11 PHP Functions - Returning values

### **Example**

<?php declare(strict\_types=1); // strict requirement  
function sum(int $x, int $y) {  
  $z = $x + $y;  
  return $z;  
}  
  
echo "5 + 10 = " . sum(5, 10) . "<br>";  
echo "7 + 13 = " . sum(7, 13) . "<br>";  
echo "2 + 4 = " . sum(2, 4);  
?>

