Taks 2

Search on constants types

First of all, constants are automatically global across the entire script.
To create a constant use the define() function. Syntax define(name, value, case-insensitive)
parameters:

name: Specifies the name of the constant
value: Specifies the value of the constant
case-insensitive: Specifies whether the constant name should be case-insensitive. The default is false.

```
<?php

// case-sensitive constant name
define("warm welcome", "welcome everyone!");
echo warm welcome;

?>
```

```
<?php
// case-insensitive constant name
define("GREETING", "Welcome to W3Schools.com!", true);
echo Greeting;
?>
```

By setting the third parameter to **true**, you allow the constant name "GREETING" to be used in a case-insensitive manner. this means that you can reference the constant as "GREETING," "greeting," "Greeting," or any other combination of upper and lower case letters, and PHP will treat it as the same constant.

Search for truthy and falsy values

Truthy Values in PHP:

In PHP a value is considered "**truthy**" if it evaluates to true in a Boolean context. Truthy values include, but are not limited to:

1- Non-empty Strings: anything at least one character is considered truthy
2- Non-empty numbers: any number other than '0' is truthy
3- Arrays: non-empty arrays: are considered truthy

Falsy Values in PHP:

Conversely, a value is considered "falsy" if it evaluates to false in a Boolean context. Falsy values include, but are not limited to:

1- Empty strings: an empty string (" ") is considered falsy
2- Zero(0): the integer '0' is falsy
3- Null: the 'null' Value is considered falsy
4- Empty Arrays: an empty array ([ ]) is considered falsy
5- False: the Boolean value 'false' is, of course falsy. XD

Truthy and falsy

```
$value = 42;

if ($value) {
  echo "Value is truthy";
} else {
   echo "Value is falsy";
}
```

if $value is any non-zero number it will be considered truthy and the "Value is **truthy**" message will be printed. If $value is **0**, it will be considered **falsy**, and the "Value is **falsy**" message will be **printed**.

search on PHP Arrays

An array stores multiple values in one single variable, a special variable, that can hold more than one value at a time.

Create an Array in PHP

in PHP, the array() Function is used to create an array:

array();

in PHP, there are three types of array:

Indexed arrays – arrays with a numeric index
Associative arrays – Arrays with named keys
Multidimensional arrays – arrays containing one or more arrays
Sorting Arrays - The elements in an array can be sorted in alphabetical or numerical order, descending or ascending.

Get The Length of an Array - The count() Function

The count() function is used to return the length (the number of elements) of an array:

Example

```
<?php
$fruits = array(“apple”, “mango”, “blueberry”);
echo count($fruits);
?>
```

the output will be “3”

PHP “indexed array” *Two ways*

-The index can be assigned automatically (the index always starts at 0), like this:

```
$cars = array("Volvo", "BMW", "Toyota");
```

-or the index can be assigned manually:

```
$cars[0] = "Volvo";
```

```
$cars[1] = "BMW";
$cars[2] = "Toyota";
```

PHP “Associative Arrays”

Associative arrays are arrays that use named keys that you assign to them. There are **two** ways to create an associative array:

```
$age = array("mahmoud"=>"24", "Ali"=>"20", "Ezzat"=>"23");
echo "Mahmoud is ". $age['Mahmoud] . " years old.";
```

or

```
$age[Mahmoud] = "24";
$age[Ali] = "20";
$age[Ezzat] = "23";
echo "mahmoud is ". $age['Mahmoud] . " years old.";
```

PHP “Multidimensional Array”

A multidimensional array is an array containing one or more arrays. PHP supports multidimensional arrays of deep levels of two, three, four, five, or more. However, arrays more than three levels deep are hard to manage for most people.

PHP “Sorting Arrays”

The elements in an array can be sorted in alphabetical or numerical order, descending or ascending.