college CDI

Server side technologies (TCS)

Lesson plan

Class 01

PHP language

- Client side
- Static vs dynamic documents
- How HTML documents are generated by PHP
- PHP module needs to be installed and running on server
- to work offline = virtual PHP server needed:
  EasyPHP (Windows), wampserver (Windows), mamp (Apple), BigApache, etc.

PHP syntax

- **<?php** some PHP coding **?>**
- **<?php**
  print ("Hello World"**);** *// print can take only one parameter*
  ?>
- **<?php**
  echo "Hello World !"**;** *// each statement ends with a semicolon*
  ?>
- **<?php**
  echo "Hello"**,**" World!"**;** *// echo can take multiple parameters*
  ?>

Special characters

Code	**Result**
\n	New line
\r	Return
\t	Tab
\\	Backslash
\$	$
\"	Double quotes

-

Comments

-
  ```
  <?php
  // This is a single line comment

  /* This is
  a multiple lines
  comment */
  ?>
  ```

Variables

- Letters and numbers + symbols (e.g.: underscore)
- Never starts with a number
- Never includes spaces
- Case sensitive

Declaring variables

-
  ```
  <?php
  $myVariable = "Hello world!";   // string
  $x = 5;                         // integer
  $y = 7.5;                       // float
  ?>
  ```

Output data to screen

-
  ```
  <?php
  $name = "John";
  echo "Hello ";          // outputs the word « Hello »
  echo $name;             // outputs the value of the variable (John)
  echo " !";              // outputs the character « ! »
  ?>
  ```

  OR

  ```
  <?php
  $name = "John";
  print "<p>Hello ";      // both commands can output HTML tags
  var_dump($name);        // string(4) "John"
  print " !</p>";
  ?>
  ```

Concatenation

- **Concatenating variables**

```
<?php
$a = "Hello ";
$b = "world!";
echo $a . $b;
?>
```

- **Concatenating variables and strings**

```
<?php
$firstName = "John";
$lastName = "Smith";
echo "Hello " . $firstName . " " . $lastName. " !";
?>
```

- **Adding text to a string using « .= »**

```
<?php
$a = "John";
$a .= " Smith";
echo $a;
?>
```

- Using single and/or double quotes

Useful string functions

strtolower :	Converts string characters to lower case.
strtoupper :	Converts string characters to upper case.
	echo strtoupper("hello world!");
strlen :	Counts and returns a string's number of characters (including spaces).
str_word_count :	Counts and returns the number of words of a string.
substr :	Return part of the string using three parameters : the string to be shortened, the position of the starting point, the number of characters to be returned.
	echo substr($myVariable,5,16);
str_replace :	Replaces a specified string values in a given string using three parameters : the text to be replaced, the replacement text and the text that is analyzed.
	echo str_replace ("This", "Here", "This is an example.");
strpos :	Locates and return the position of a suite of character(s) within a string.
	echo strpos($myVariable,"ex");
ucfirst :	Makes the first character of a string an upper case.

lcfirst :	Makes the first character of a string an upper case.
wordwrap :	Wraps a string to a given number of characters.

```
nl2br();
$a = "An example: I wonder how it will look like using word wrap function on this text...";
echo wordwrap($a,15,"<br>\n");
```

Other functions

chop()	Removes whitespace or other characters from the right end of a string.
chr()	Returns a character from a specified ASCII value.
ltrim()	Removes whitespace or other characters from the left side of a string.
rtrim()	Removes whitespace or other characters from the right side of a string.
str_shuffle()	Randomly shuffles all characters in a string.
strcasecmp()	Compares two strings (case-insensitive).
strcmp()	Compares two strings (case-sensitive).
strrev()	Reverses a string.
trim()	Removes whitespace or other characters from both sides of a string.

Assignment 1 : Your first PHP script

In a HTML document, use PHP to obtain the following :

The user will write a sentence as a value of a predefined variable.

The output should look like this :

The sentence you entered is :
This is my sentence

It is composed of :

- 4 different words
- 19 characters (including spaces)

Note : *Your program should make sure the final output is in lower case with the first letter in upper case.*

Class 02

Variables (suite)

Scope of variables

- Global / local variables

```
<?php
$a=1;                   // global (default)
function hello() {
        echo $a;        // local to function (no data to output)
}
?>
```

- Global keyword

```
<?php
$a=1;                   // global (default)
$b=2;                   // global (default)
function plus() {
        global $a,$b;   // retrieves global variables
        global $result; // declares new global variable
        $result=$a+$b;  // assigns value to new variable
}
plus();                 // launches function
echo $result;           // outputs final data to screen : 3
?>
```

- Pre-defined $GLOBALS associative array

```
$a = 1;
$b = 2;
function plus() {
        $GLOBALS["b"] = $GLOBALS["a"] + $GLOBALS["b"];
}
plus();
echo $b;                //b now equals 3
?>
```

Constants

- Like variables
- Global
- Value never changed by program
- Written in UPPERCASE by convention

```
<?php
$price = 9.99;              // variable
define("TAX", 1.15);        // constant's value representing 15% tax to apply
echo $price * TAX;
?>
```

Some predefined constants

__FILE__ :	Returns the name of the file including its full path.
__DIR__ :	Returns the name of the directory of the file including its path.
__LINE__ :	Returns the line number in the source file.
	echo "<b>Full path :</b> " . __FILE__ . "<br />";

Booleans

- true (1) / false (0)

- **Boolean is *false* when the value is :**

  - 0 (zero) integer
  - 0.0 (zero) float
  - Empty or zero string (e.g. : " " or "0")
  - Empty array
  - Empty object
  - NULL special constant

  - Bolean is *true* in all other circumstances

Numbers

Integers

```
<?php
$a = 1234;            // decimal
$a = -123;            // a negative
$a = 0123;            // octal (equivalent to 83 decimal)
$a = 0x1A;            // hexadecimal (equivalent to 26 decimal)
$a = 0b11111111;      // binary (equivalent to 255 decimal)
?>
```

Operators

```
<?php
echo 10 + 2;
echo 10 - 2;
echo 10 * 2;
echo 10 / 2;
?>
```

Float

- Numbers with decimals
- Operation on integers generating decimals : result = float

  Example :
  ```
  <?php
  var_dump(41/8);
  ?>
  ```

  Result :
  ```
  float(5.125)
  ```

Converting a value into an integer

- ***(int)***, ***(intval)*** or ***(integer)***
  Converting a float to an integer rounds the value to the lowest integer (rounds down)

  ```
  <?php
  var_dump(41/8);                    // result : float(5.125)
  var_dump((int) (41/8));            // result : int(5)
  var_dump((integer) (41/8));        // result : int(5)
  var_dump((intval) (41/8));         // result : int(5)
  ?>
  ```

Numerical strings

- Numbers in string do not always behave as strings but as integers

```
<?php
echo 123;               // result : 123
echo 0123;              // result : 83 (octal numbers start with 0)
echo "0123";            // result : 0123 (string)
echo "0123" + "10";     // result : 133 (123 + 10)
echo "0123" + 10;       // result : 133 (123 + 10)
?>
```

Comparing values

- PHP have problems comparing a number value with the result from operations = Need to use the ***round()*** function.

```
<?php
$x = 8 - 6.4;                           // equals 1.6
$y = 1.6;
var_dump($x == $y);                     // boolean = false
?>

<?php
$x = 8 - 6.4;                           // equals 1.6
$y = 1.6;
var_dump(round($x) == round($y));       // boolean = true
?>
```

Booleans and numbers

- **Provide the same result :**
  if($var==true) echo "ok";
  if($var) echo "ok";

- A negative value (e.g. : -2) is considered true (it isn't 0).

- **Booleans converted into integers are :**
  False = 0
  True = 1

- **Booleans converted into floats are rounded down :**
  1.9 = 1
  0.8= 0

-

Math functions

ceil()	Rounds a number up to the nearest integer
exp()	Calculates the exponent of e
floor()	Rounds a number down to the nearest integer
lcg_value()	Returns a pseudo random number in a range between 0 and 1
is_float()	Returns true if a value is a float
is_int()	Returns true if a value is an integer
is_nan()	Checks whether a value is 'not-a-number'
is_numeric()	Returns true if a value is numerical
max()	Returns the highest value in an array, or the highest value of several specified values
min()	Returns the lowest value in an array, or the lowest value of several specified values
mt_rand()	Generates a random integer using Mersenne Twister algorithm
number_format()	Formats a number with groups of thousands
pi()	Returns the value of PI
pow()	Returns x raised to the power of y
rand()	Generates a random integer
round()	Rounds a floating-point number
sqrt()	Returns the square root of a number

Math constants

M_PI	Returns Pi
PHP_ROUND_HALF_UP	Round halves up
PHP_ROUND_HALF_DOWN	Round halves down

Date and time

```
<?php
echo "<b>Today is :</b> " . date("l M d, Y") . "<br>";
echo "<b>Today is :</b> " . date("Y/m/d") . "<br>";
echo "<b>Today is :</b> " . date("Y.m.d") . "<br>";
echo "<b>Today is :</b> " . date("Y-m-d") . "<br>";
echo "<b>Today is :</b> " . date("l");                    // Lower case «L»
?>
```

```
<?php
date_default_timezone_set("America/New_York");
echo "<b>The time is :</b> " . date("H:i:s");
?>
```

Result :
The time is : 21:34:28

```
<?php
echo "<b>The time is :</b> " . date("h:i:sa");
?>
```

Result :
The time is : 09:35:33pm

Assignment 2 : Your first PHP script

In a HTML document, use PHP to obtain the following :

The user will enter different values for the main global variables (table diameter and side a and be of a triangle) and the program will output a message such as the following :

Date :
Saturday, April 6th 2019

Time :
10:25 am

Your table details :

Diameter :	*User enters diameter*
Circumference :	*Program output*
Surface area :	*Program output*

The length of your triangle hypotenuse :

Side a :	*User enters length of side a*
Side b :	*User enters length of side b*
Hypotenuse :	*Program output*

Class 03

Arrays

Creating an array

- Array = variable containing several values
- Values = indexed with numbers starting with zero
- Many ways to create an array using ***array()***
  OR directly OR automatically assigning indexes to variables

```
<?php
$myInstruments = array("Piano", "Drum", "Guitar");
?>
```

OR

```
$myInstruments = array();
$Instruments[0] = "Piano";
$Instruments[1] = "Drum";
$Instruments[2] = "Guitar";
```

OR

```
$myInstruments = array();
$Instruments[] = "Piano";
$Instruments[] = "Drum";
$Instruments[] = "Guitar";
```

Output to screen :

```
<?php
echo $Instruments[0];                    // result : Piano
?>
```

OR

```
<?php
echo "I like hearing a " . $Instruments[1] . "and a " . $Instruments[2]
// result : I like hearing a Drum and a Guitar
?>
```

Replacing index numbers with strings

```
<?php
$myInstruments = array();
$Instruments[Chopin_favorite] = "Piano";
$Instruments[Boom_Boom] = "Drum";
$Instruments[Hendrix_thing] = "Guitar";

echo $Instruments[Hendrix_thing];          // would show : Guitar
?>
```

Creating an array using keys

Syntax :

```
<?php
$myArray = array("a"=>"Piano", "name"=>"John", 29, "Karen");
?>
```

Outputs to screen :

```
echo $myArray["a"];              // would show : Piano
echo $myArray["name"];           // would show : John
echo $myArray[0];                // would show : 29
echo $myArray[1];                // would show : Karen
```

Index are attributed to «unkeyed» values the regular way :

```
<?php
$myArray = array("a"=>"Piano", 29, "name"=>"John", "Karen");
                              [0]                  [1]
?>
```

Conditions

Control operators

==	Equal (type and value)
!=	Different of
<	Lower than
>	Higher than
<=	Lower or equal to
>=	Higer or equal to
and OR **&&**	AND / AND (type and value)
or OR **||**	OR / OR (type and value)

If, else, elseif

```
<?php
$myNumber = 11;

if ($myNumber >= 0 && $myNumber < 10) {          // if the value of the variable is between 0 and 9 :
        echo $myNumber . " is between 0 and 9";
}

elseif ($myNumber >= 10 && $myNumber < 20) {     // if not, if the value of the variable between 10 and 19 :
        echo $myNumber . " is between 10 and 19";
}

else {                                           // if none of the two preceding conditions apply :
        echo $myNumber . " is higher than 19";
}
?>
```

Explanation

- FIRST : $myNumber is tested (with the condition between parenthesis)
  Equal or higher than zero AND lower than 10
  TRUE : echo « 11 is between 0 and 9 »

- NOW 2 possibilities :
  else = directly echo an appropriate message OR second condition to test if the first one fails (***elseif***)
  $myNumber tested : equals or is higher than 10 OR lower than 20
  TRUE : echo « 11 is between 10 and 19 »

- FINALLY : if both conditions false
  else is executed : echo « 11 is higher than 19 ».

Switch

Equivalent of ***if*** followed by several ***elseif***

```
<?php
$name = "Morpheus";

switch ($name) {
        case "John" :                       // if John
        echo "Your name is John.";          // output John
        break;

        case "Karen" :
        echo "Your name is Karen.";
        break;

        case "Morpheus" :
        echo "Your name is Morpheus.";
        break;

        default :                           // if none of the possibilities
        echo "I don't know who you are.";   // output default message
}
?>
```

Comparison with *if* structure:

```
<?php
$name = "Morpheus";

if ($name == "John") {                      // if John
        echo "Your name is John.";          // output John
}

elseif ($name == "Karen") {                 // not John but Karen
        echo "Your name is Karen.";         // output Karen
}

elseif ($name == "Morpheus") {              // not Karen but Morpheus
        echo "Your name is Morpheus.";
}

else {                                      // otherwise
        echo "I don't know who you are.";   // output this message
}
?>
```

While

The while condition structure allows us to create a loop repeating itself as long as a condition is true.

Syntax :

```
<?php
$number = 5;
$i = 0;
while ($i < $number) {
        echo "Our number isn't " . $i. "<br />";
        $i = $i + 1;                              // OR $i++;
}
echo "Our number is ". $i;
?>
```

Result :

```
Our number isn't 0
Our number isn't 1
Our number isn't 2
Our number isn't 3
Our number isn't 4
Our number is 5
```

Explanation :

- Value of ***$number*** = 5
  iteration variable (***$i***) = 0
- *While* creates a loop (it repeats itself) as long as the condition is ***true***
  (***as long as $i is lower than 5***)
- When the condition becomes ***false*** (***when $i is equal or higher than 5***)
  the loop is stopped and the program continues being executed (next commands)

for

- Allows to create loops repeating itself as long as a condition is true
- Useful to count or to limit number of tries, for instance
- Important structure which can replace ***if*** in many cases.

Syntax :

```
for ($i=0; $i < $number; $i++) {
        Some commands...
}
```

Explanation :

- FIRST : iteration variable (***$i***) = 0

- SECOND : condition tests if ***$i*** is lower than ***$number***
  As long as the condition is ***true***, $i is increased by 1 AND the command lines are executed

- FINALLY : When the condition is ***false***, the loop stops
  AND the rest of the program is then executed

Example :

```
<?php
$number = 5;
for ($i=1; $i < $number; $i++) {
        echo "The number is " . $number . "<br />";
}
echo "And it has been output " . $i . " times";
?>
```

Result :

```
The number is 10
The number is 10
The number is 10
The number is 10
The number is 10
And it has been output 5 times
```

Explanation :

- ***$number*** = *5* AND **$i** = *0*

- Condition : *as long as* ***$i is lower than 10***) commands are executed

- When the condition becomes *false* (when ***$i is equal or higher than 10***), loop stops and program continues

Assignment 3 : Guessing the secret number

In a HTML document, use PHP to obtain the following :

The first variable ($number) to be declared is the one representing the number chosen by the user.

The second variable declared ($secret) is the secret number. You need for the program to generate a random number between 0 and 10.

You need messages saying whether the secret number is higher, lower or equal to the chosen number.