B2 - Reboost

B-RBT-200

Backend

PHP Introduction

1.1

{EPITECH.}

Backend

repository name: RBT_back_d01_$ACADEMICYEAR
repository rights: anthony1.palasse@epitech.eu fabien-luc1.lallemand@epitech.eu kevin.cazal@epitech.eu nicolas1.ah-leung@epitech.eu william.grondin@epitech.eu

- Your repository must contain the totality of your source files, but no useless files (binary, temp files, obj files,...).
- Error messages have to be written on the error output, and the program should then exit with the 84 error code (0 if there is no error).

Foreword

Today, you will learn the first bricks of PHP programming. You will find it easier than C programming. No more types but DO NOT forget semi-colons.

Please note that prototypes are just here as information. They are deprecated in PHP5 and newer.

Tasks

+ Task 01

Turn in: /task_01/task_01.php
Restrictions: None.

Create the variables "integer", "float", "string", "bool", "null". Give the following values to the variables whose name correspond to the type of value: "true", "forty two", "42", "NULL" and "42.42".

+ Task 02

Turn in: /task_02/task_02.php
Restrictions: None.

Create a variable named "var" and set it to "Hello World".
Print the content of this variable.
Then delete the variable "var".
Now try to print the content of this variable again.

+ Task 03

Turn in: /task_03/task_03.php
Restrictions: None.

Create an array named "my_array" which will contain 5 elements of type (in that order): "string", "integer", "string", "float" and "string" These elements will have the following values, respectively: "to", 42, "Glory", "42.42" and "Geckos".

+ TASK 04

Turn in: /task_04/task_04.php
Prototype: void my_concat(mixed $str1, mixed $str2);
Restrictions: You will have to choose between "echo" and "print" for this function, and you will use only once the display function that you have chosen.

Create a function named "my_concat" that takes two parameters. The function must display the first parameter followed by a space followed by the second parameter.

Example:

Terminal

```
~/B-RBT-200> cat index.php
<?php
require("task_04.php");
my_concat("Hello", "world");
my_concat("My name is", "Odin");
```

Terminal

```
~/B-RBT-200> php index.php
Hello worldMyname is Odin
```

+ TASK 05

Turn in: /task_05/task_05.php
Prototype: void my_swap(mixed &$a, mixed &$b);
Restrictions: None.

Write a function that exchanges the content of two variables whose **references** are given as parameters.

+ TASK 06

Turn in: /task_06/task_06.php
Prototype: string get_angry_dog(int $nbr);
Restrictions: Obligation to use the keyword "for".

Write a function that returns a string composed of as many "woof" as the value of the variable passed as parameter, followed by a new line.
Example:

```
Terminal
~/B-RBT-200> cat index.php
<?php
require("task_06.php");
echo (get_angry_dog(3));
```

```
Terminal
~/B-RBT-200> php index.php
woofwoofwoof
~/B-RBT-200>
```

+ TASK 07

Turn in: /task_07/task_07.php
Prototype: void print_array(array $my_array);
Restrictions: Obligation to use the keyword "foreach".

Write a function that displays all the values of the array passed as parameter, each value being followed by a new line.

+ TASK 08

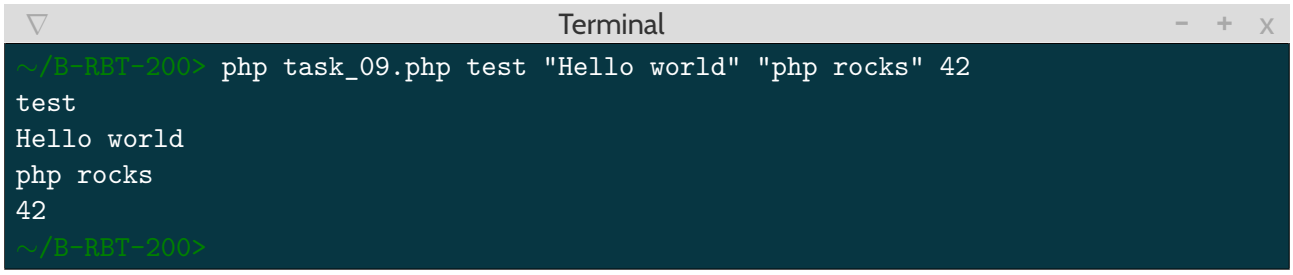
Turn in: /task_08/task_08.php
Prototype: array my_get_args([mixed $var, ...]);
Restrictions: Use of func_get_args forbidden.

Write a function my_get_args that takes as parameter a variable number of arguments and returns these arguments in an array.

+ TASK 09

Turn in: /task_09/task_09.php
Prototype: void my_print_args([mixed $var, ...]);

Create a function that prints each arguments passed to the script followed by a new line.


Terminal
~/B-RBT-200> php task_09.php test "Hello world" "php rocks" 42
test
Hello world
php rocks
42
~/B-RBT-200>

+ TASK 10

Turn in: task_10/task_10_1.php
task_10/task_10_2.php
Prototype: my_included_putstr(mixed $str);
Restrictions: None.

In task_01_2.php, create a function, my_included_putstr which will be an echo of the string given as parameter.
In task_01_1.php, include the task_01_2.php and call the function my_included_putstr giving it the first argument from command line.

+ TASK 11

Turn in: task_11/task_11.php

After a hacker attack, the Facebook database was compromised. We have determined that the hashed password of the user named "admin" is: "21232f297a57a5a743894a0e4a801fc3".
Create a file named "task_11.php" and write inside it the password followed by a newline.

The algorithm used is MD5

+ TASK 04

Turn in: task_12/task_12.php
Prototype: array my_password_hash(string $password);
Prototype: bool my_password_verify(string $password, string $salt, string $hash);

Write a function my_password_hash that takes as parameter a password in plain-text and returns an array containing the hashed password in MD5, as well as the "salt" used for hashing. The salt must be different on each call.
Then write a function my_password_verify that takes as parameter a password in plain-text, a salt and a hashed password. This function must return true or false, depending on whether the hashed password corresponds to the plain-text password with the associated salt or not.
The array returned by the function my_password_hash has to look like this:

Terminal

```
$array = [
     "hash" => "XXXXX",
     "salt" => "12345",
 ];
```

+ BONUS

Turn in: bonus/bonus.php
Prototype: void sequence(int $nbr);

Write a function that outputs this sequence to the nth iteration.
Caution: first iteration is 0.
Your function should not print anything if you pass it a negative number.

Example:

Terminal

```
<?php
require("bonus.php");
sequence(5);
```

Produce the following output :

Terminal

```
1
11
21
1211
111221
312211
```