PHP – Bases

Call me by my_name()

Introduction

Ce sujet a pour but de vous faire reprendre les bases et de vous remettre en jambe sur PHP et sa syntaxe, en vous faisant travailler avec des fonctions.
Comme vous allez apprendre à faire des fonctions, **aucune fonction de base ou bibliothèque de PHP n'est autorisée à part var_dump() et isset().**
Dans un dossier **runtrack-b2-php**, créez un dossier **jour-01**. Dans ce dossier, créez un dossier **job-XX** pour chacun des jobs **où XX correspond au numéro de chaque job.**
Pensez à faire des commits réguliers et à **respecter les consignes dans le sujet**.

Votre code doit juste s'exécuter correctement. Vous pouvez bien évidemment utiliser var_dump() pour le débugger, mais pensez à clean votre code avant de rendre votre projet.

Bon courage !

Job 1

Dans le dossier **job-01**, faites un fichier index.php. À l'intérieur de ce fichier index.php, ajouter une fonction **my_str_search()**. Cette fonction permettra de *compter le nombre d'occurrences d'une lettre dans une chaîne de caractères*.

```php
function my_str_search(string $haystack, string $needle) : int {
  /**
   * Your code here
   */
}
```

Voici la signature de la fonction, **veillez à la respecter :**
Elle prendra en premier paramètre la lettre à chercher dans la chaine de caractère, et en deuxième paramètre la chaine de caractères dans laquelle chercher. Cette fonction retourne un entier, le nombre d'occurrences de la lettre.

```php
my_str_search('La Plateforme', 'a') === 2;
```

Job 2

Dans le dossier **job-02**, faites un fichier index.php. À l'intérieur de ce fichier index.php, ajouter une fonction **my_str_reverse()**. Cette fonction permettra d'*inverser l'ordre des lettres d'une chaîne de caractères*.

```php
function my_str_reverse(string $string) : string {
  /**
   * Your code here
   */
}
```

Voici la signature de la fonction, **veillez à la respecter :**

```php
my_str_reverse('Hello') === 'olleH';
```

Elle prendra en premier paramètre la chaine de caractères à inverser. Cette fonction retourne la chaîne de caractère inversée.

Job 3

Dans le dossier job-03, faites un fichier index.php. À l'intérieur de ce fichier index.php, ajouter une fonction **my_is_multiple()**. Cette fonction permettra de *déterminer si un nombre est un multiple d'un autre*.

Voici la signature de la fonction, **veillez à la respecter :**

```php
function my_is_multiple(int $divider, int $multiple) : bool {
  /**
   * Your code here
   */
}
```

```php
my_is_multiple(2, 4) === true;

my_is_multiple(2, 5) === false;
```

Job 4

Dans le dossier job-04, faites un fichier index.php. À l'intérieur de ce fichier index.php, faites une fonction **my_fizz_buzz()**. Cette fonction retournera un tableau de la longueur entrée en paramètre, avec une structure donnée. Le tableau retourné devra suivre les règles suivantes :

- Des entiers dans l'ordre croissant, allant de 1 à la longueur du tableau
- Si la valeur est un multiple de **3**, elle est remplacée par **Fizz**
- Si la valeur est un multiple de **5**, elle est remplacée par **Buzz**
- Si c'est un multiple de **3 et 5**, elle est remplacée par **FizzBuzz**

Voici la signature de la fonction, **veillez à la respecter :**

```php
function my_fizz_buzz(int $length) : array {
  /**
   * Your code here
   */
}
```

```
my_fizz_buzz(15) === [1, 2, 'Fizz', 4, 'Buzz', 'Fizz', 7, 8, 'Fizz', 'Buzz', 11, 'Fizz', 13, 14, 'FizzBuzz'];
```

Job 5

Dans le dossier job-05, faites un fichier index.php. À l'intérieur de ce fichier index.php, faites une fonction **my_is_prime()**. Cette fonction permettra de déterminer si un nombre est premier. Pour rappel, un nombre premier est un nombre divisible uniquement par lui-même. On dit qu'un entier *a* est divisible par un entier *b* s'il existe un entier *k* tel que *a = bk*.

Voici la signature de la fonction, **veillez à la respecter :**

```php
function my_is_prime(int $number) : bool {
  /**
   * Your code here
   */
}
```

```php
my_is_prime(3) === true;

my_is_prime(12) === false;
```

Job 6

Dans le dossier job-06, faites un fichier index.php. À l'intérieur de ce fichier index.php, faites une fonction **my_array_sort()**. Cette fonction permettra de trier un tableau. Ce tri doit être possible dans l'ordre croissant ou décroissant. Le deuxième paramètre de cette fonction ne peut prendre que deux valeurs : "ASC" ou "DESC".

Voici la signature de la fonction, **veillez à la respecter :**

```php
function my_array_sort(array $arrayToSort, string $order) : array {
  /**
   * Your code here
   */
}
```

```php
my_array_sort([2, 24, 12, 7, 34], 'ASC') === [2, 7, 12, 24, 34];

my_array_sort([8, 5, 23, 89, 6, 10], 'DESC') === [89, 23, 10, 8, 6, 5];
```

Bonus : Ajoutez la possibilité de trier des chaînes de caractères par ordre alphabétique croissant et décroissant.

Job 7

Dans le dossier job-07, faites un fichier index.php. À l'intérieur de ce fichier index.php, faites une fonction **my_clossest_to_zero()**. Cette fonction doit permettre de **récupérer le nombre le plus proche de 0 parmi une liste de nombre**. Ces nombres sont des entiers **positifs ou négatifs**.

Voici la signature de la fonction, **veillez à la respecter :**

```php
function my_closest_to_zero(array $array) : int {
  /**
   * Your code here
   */
}
```

```php
my_closest_to_zero([2, -1, 5, 23, 21, 9]) === -1;

my_closest_to_zero([234, -142, 512, 1223, 451, -59]) === -59;
```

Compétences visées

- ➔ Algorithmie
- ➔ PHP

Rendu

Le projet est à rendre sur **https://github.com/prenom-nom/runtrack-b2-php**.

C'est le premier jour de RunTrack, donc normalement, vous n'avez que le dossier jour-01 à push. Faites attention à bien **respecter la signature de chacune de vos fonctions** et les **noms de fichiers et de dossiers à la lettre !**

Base de connaissances

➔ PHP