W1- Piscine PHP

W-WEB-024

Jour 07

Recusivité

v1.0

EPITECH.
L'ECOLE DE L'INNOVATION ET DE L'EXPERTISE INFORMATIQUE

Informations

Avant de commencer

- Lisez attentivement toutes les consignes.
- Consultez vos mails plusieurs fois par jour, tous les jours.

Commencez par lire vos mails tout de suite à l'adresse : mail.office365.com.

- C'est une pangolinette (un programme) qui corrige vos exercices. Vérifiez le travail que vous allez rendre afin qu'il respecte scrupuleusement les consignes
- Vous devez respecter les restrictions qui sont imposées dans chaque exercice. Le cas contraire, la pangolinette va considérer comme **triche** en attribuant la note de -42
- Vous êtes susceptibles à tout moment de recevoir des corrections intermédiaires.

Pour bénéficier de corrections intermédiaires, vous devez chaque jour :

- Etre inscrit au projet et aux activités dans l'intranet.
- Avoir créé le dépôt avec BLIH.
- Tenir à jour régulièrement le dépôt.

- Ne laissez jamais votre session ouverte sans surveillance.

Jour 07

Recusivité

Nom du répertoire:	Piscine_PHP_Jour_07
Droits de ramassage:	ramassage-tek
langage:	php
Taille du groupe:	1

- Votre repertoire ne doit pas contenir de fichiers inutiles (fichiers temporaires, ...)
- Vous ne devez pas oublier votre fichier *auteur*, si vous l'oubliez, la moulinette ne pourra pas vous corriger.
- N'oubliez pas de push régulièrement vos fichiers, sans cela, pas de correction.

Pensez à créer votre répertoire en début de journée et à envoyer votre travail via **git** !
Le nom du répertoire est spécifié dans les instructions pour chaque étape / exercice.
Pour garder votre répertoire propre, regardez du côté de `gitignore`.

N'oubliez pas de vous inscrire à toutes les activités possible de la semaine.

Etape 1

2 points

Nom de rendu: `Piscine_PHP_Jour_07/ex_01.php`
Restrictions: Aucune

Écrire une fonction itérative qui retourne le résultat de l'opération factorielle à partir du nombre passé en paramètre. En cas d'erreur, si les arguments ne sont pas du bon type ou si la fonction est appelée sans argument, la fonction devra retourner NULL.

Prototype: mixed my_facto(int $nbr);

0! = 1

(-564)! = NULL

Etape 2

2 points

Nom de rendu: `Piscine_PHP_Jour_07/ex_02.php`
Restrictions: Aucune

Écrire une fonction similaire à la précédente mais de façon récursive.

Prototype: mixed my_facto_rec(int $nbr);

Etape 3

2 points

Nom de rendu: `Piscine_PHP_Jour_07/ex_03.php`
Restrictions: la fonction pow() est interdite

Écrire une fonction itérative qui retourne le premier nombre passé en paramètre élevé à la puissance du deuxième nombre passé en paramètre.
En cas d'erreur, si les arguments ne sont pas du bon type ou si la fonction est appelée sans argument, la fonction devra retourner NULL.
Les nombres entiers négatifs ou nul ne sont pas des erreurs pour $nb_a. Par contre, pour des raisons de simplicité dans notre exercice, les nombres entiers négatifs sont des erreurs pour $nb_b.

Prototype: mixed my_pow(int $nb_a, int $nb_b);

0 ^0 = 1

n ^0 = 1

(-564) ^1 = (-564)

Etape 4

2 points

Nom de rendu: `Piscine_PHP_Jour_07/ex_04.php`
Restrictions: la fonction pow() est interdite

Écrire une fonction similaire à la précédente mais de façon récursive.

Prototype: mixed my_pow_rec(int $nbr, int $power);

Etape 5

2 points

Nom de rendu: `Piscine_PHP_Jour_07/ex_05.php`
Restrictions: Aucune

Écrire une fonction itérative qui retourne le résultat des additions du premier nombre passé en paramètre avec le deuxième.
À chaque addition, la valeur absolue du deuxième nombre sera décrémenté, et ce jusqu'à 0.
En cas d'erreur, si les arguments ne sont pas du bon type ou si la fonction est appelée sans argument, la fonction devra retourner NULL.

Prototype: mixed sum_it(int $nbr, int $iteration);
Exemple:

```
sum_it(5, 3);
// retourne 11 (car 5 + 3 + 2 + 1 = 11)
sum_it(5, -3);
// retourne -1 (car 5 - 3 - 2 - 1 = -1)
sum_it(-5, -3);
// retourne -11 (car -5 - 3 - 2 - 1 = -11)
```

Etape 6

2 points

Nom de rendu: `Piscine_PHP_Jour_07/ex_06.php`
Restrictions: Aucune

Écrire une fonction similaire à la précédente mais de façon récursive.

Prototype: mixed sum_rec(int $nbr, int $iteration);

Etape 7

2 points

Nom de rendu: `Piscine_PHP_Jour_07/ex_07.php`
Restrictions: Aucune

Écrire une fonction qui prend en paramètres une chaîne de caractère et un délimiteur.
Elle doit retourner un tableau contenant les morceaux de la chaîne découpée grâce au délimiteur.
En cas d'erreur, si les arguments ne sont pas du bon type ou si la fonction est appelée sans argument, la fonction devra retourner NULL.

Prototype: mixed str_to_wordtab(string $str, string $delim);
Exemple:

```
str_to_wordtab("Mais tu es tout la", ' ');
// retourne array("Mais", "tu", "es", "tout", "la")
```

Etape 8

2 points

Nom de rendu: `Piscine_PHP_Jour_07/ex_08.php`
Restrictions: Aucune

Écrire une fonction qui prend en paramètre une chaîne de caractères.
Elle doit la retourner avec pas plus d'un espace consécutif entre chaque mot, ni aucun espace au début et à la fin, le tout avec une majuscule à chaque fin de mot uniquement.
En cas d'erreur, si les arguments ne sont pas du bon type ou si la fonction est appelée sans argument, la fonction devra retourner NULL.

Prototype: mixed str_beautifuler(string $str);
Exemple:

```
str\_beautifuler("Un PanGolin CacHe ");
// retourne "uN pangoliN cachE"
```

Etape 9

2 points

Nom de rendu: `Piscine_PHP_Jour_07/ex_09.php`
Restrictions: Aucune

Écrire une fonction qui teste si le nombre passé en paramètre est un nombre premier et qui renvoie true ou false.
En cas d'erreur, si les arguments ne sont pas du bon type ou si la fonction est appelée sans argument, la fonction devra retourner NULL.

Prototype: mixed is_prime(int $nbr);

Etape 10

2 points

Nom de rendu: `Piscine_PHP_Jour_07/ex_10.php`
Restrictions: Aucune

Écrire une fonction qui retourne le nombre premier immédiatement supérieur ou égal au nombre passé en paramètre. En cas d'erreur, si les arguments ne sont pas du bon type ou si la fonction est appelée sans argument, la fonction devra retourner NULL.

Prototype: mixed get_next_prime(int $nbr);
Exemple:

```
get_next_prime(8);
// retourne 11
```