



# W1 - Piscine PHP

---

W-WEB-024

## Jour 07

---

Récurtivité



## Jour 07

repository name: poolphpday07  
language: PHP



- The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.

## 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](mailto: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.
- Tenir à jour régulièrement le dépôt.

- Ne laissez jamais votre session ouverte sans surveillance.

## JOUR 07

---



Afin de devenir de bons développeurs il vous faut être rigoureux et méthodique. Cela s'applique aussi à la qualité du code que vous allez produire. Pour cela, vous devez respecter des normes de code ! Pour le PHP, par exemple, vous devez respecter la norme [PSR-12](#).



Toute erreur de norme entraînera un malus conséquent sur votre note finale.



Votre répertoire ne doit pas contenir de fichiers inutiles (fichiers temporaires, ...). 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 possibles de la semaine.



## ÉTAPE 1 - (2 POINTS)

---

Nom de rendu : 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`

## ÉTAPE 2 - (2 POINTS)

---

Nom de rendu : ex\_02.php

Restrictions : Aucune

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

**Prototype** : `mixed my_facto_rec(int $nbr);`



### ÉTAPE 3 - (2 POINTS)

---

Nom de rendu : ex\_O3.php

Restrictions : La fonction pow() est interdite. L'opérateur \*\* est interdit

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 nuls 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);



$$0 ^ 0 = 1$$



$$n ^ 0 = 1$$



$$(-564) ^ 1 = (-564)$$

### ÉTAPE 4 - (2 POINTS)

---

Nom de rendu : ex\_O4.php

Restrictions : La fonction pow() est interdite. L'opérateur \*\* est interdit

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

Prototype : mixed my\_pow\_rec(int \$nbr, int \$power);



## ÉTAPE 5 - (2 POINTS)

---

Nom de rendu : ex\_O5.php

Restrictions : Aucune

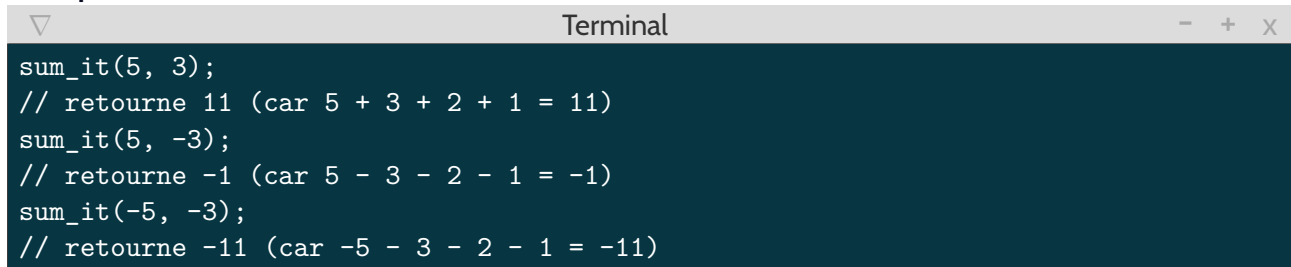
Écrire une fonction itérative qui retourne le résultat des additions du premier nombre passé en paramètre avec le deuxième.

À chaque addition, la valeur absolue du deuxième nombre sera décrémentée, 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)
```

## ÉTAPE 6 - (2 POINTS)

---

Nom de rendu : ex\_O6.php

Restrictions : Aucune

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

**Prototype :** mixed sum\_rec(int \$nbr, int \$iteration);

## ÉTAPE 7 - (2 POINTS)

Nom de rendu : ex\_07.php

Restrictions : Aucune

Écrire une fonction qui prend en paramètres une chaîne de caractères 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** :

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



explode

## ÉTAPE 8 - (2 POINTS)

Nom de rendu : 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** :

```
Terminal
str_beautifuler("Un PanGolin CacHe");
// retourne "uN pangoliN cachE"
```





## ÉTAPE 9 - (2 POINTS)

---

Nom de rendu : 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);

## ÉTAPE 10 - (2 POINTS)

---

Nom de rendu : 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 :**

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