



<https://github.com/baduritsko/algogymfr>

GNU General Public License v3.0

Version 7

# Série 1

Boucle for



# 1a - Ligne verticale

Langages : Python ou PHP (débutant)

En entrée : une valeur numérique donnée par l'utilisateur.

Le programme affiche une ligne verticale de caractères.

```
Indiquer une valeur : 5
```

```
*  
*  
*  
*  
*
```

# 1b - Ligne horizontale

Langages : Python ou PHP (débutant)

En entrée : une valeur numérique donnée par l'utilisateur.

Le programme affiche une ligne horizontale de caractères.

```
Indiquer une valeur : 5  
*****
```

# Série 2

Boucle for – condition if



## 2a - Triangle rectangle – bas / gauche

Langages : Python ou PHP (débutant)

En entrée : une valeur numérique donnée par l'utilisateur.

Le programme affiche un triangle rectangle avec l'angle droit en bas à gauche.

Indiquer une valeur : 5

```
*  
**  
***  
****  
*****
```

## 2b - Triangle rectangle – haut / gauche

Langages : Python ou PHP (débutant)

En entrée : une valeur numérique donnée par l'utilisateur.

Le programme affiche un triangle rectangle avec l'angle droit en haut à gauche.

```
Indiquer une valeur : 5
```

```
*****
```

```
****
```

```
***
```

```
**
```

```
*
```

## 2c - Triangle rectangle – haut / droite

Langages : Python ou PHP (débutant)

En entrée : une valeur numérique donnée par l'utilisateur.

Le programme affiche un triangle rectangle avec l'angle droit en haut à droite.

Indiquer une valeur : 5

\*\*\*\*\*

\*\*\*\*

\*\*\*

\*\*

\*



# 2d - Triangle rectangle – bas / droite

Langages : Python ou PHP (débutant)

En entrée : une valeur numérique donnée par l'utilisateur.

Le programme affiche un triangle rectangle avec l'angle droit en bas à droite.

Indiquer une valeur : 5

```
*  
**  
***  
****  
*****
```

# Série 3

Boucle for



# 3a - Triangle isocèle vers le haut

Langages : Python ou PHP (débutant)

En entrée : une valeur numérique donnée par l'utilisateur.

Le programme affiche un triangle isocèle vers le haut.

Indiquer une valeur : 5

```
*  
***  
*****  
*****  
*****
```

## 3b - Triangle isocèle vers le bas

Langages : Python ou PHP (débutant)

En entrée : une valeur numérique donnée par l'utilisateur.

Le programme affiche un triangle isocèle vers le bas.

```
Indiquer une valeur : 5
```

```
*****
```

```
*****
```

```
*****
```

```
***
```

```
*
```

## 3c - Losange plein

Langages : Python ou PHP (débutant)

En entrée : une valeur numérique donnée par l'utilisateur.

Le programme affiche un losange.

Indiquer une valeur : 5

```
*  
***  
*****  
***  
*
```

# 3d – Sapin de Noël

Langages : Python ou PHP (débutant)

En entrée : une valeur numérique donnée par l'utilisateur.

Le programme affiche des triangles de plus en plus grand avec un décalage

Indiquer une valeur : 5

```
  *
  *
 ***
  *
 ***
 *****
  *
 ***
 *****
 *****
  *
 ***
 *****
 *****
 *****
```

# Série 4

Boucle for



## 4a - Triangle isocèle inverse vers le haut

Langages : Python ou PHP (débutant)

En entrée : une valeur numérique donnée par l'utilisateur.

Le programme affiche un triangle isocèle vers le haut.

```
Indiquer une valeur : 5
```

```
*****
```

```
***   ***
```

```
**     **
```

```
*       *
```



## 4b - Triangle isocèle inverse vers le bas

Langages : Python ou PHP (débutant)

En entrée : une valeur numérique donnée par l'utilisateur.

Le programme affiche un triangle isocèle vers le bas.

Indiquer une valeur : 5

```
*      *  
**     **  
***    ***  
****   ****
```

# Série 5

Boucle for



# 5a - Barre oblique – haut / gauche

Langages : Python ou PHP (débutant)

En entrée : une valeur numérique donnée par l'utilisateur.

Le programme affiche une barre oblique qui commence en haut à gauche.

```
Indiquer une valeur : 5
```

```
*
```

```
 *
```

```
  *
```

```
   *
```

```
    *
```

## 5b - Barre oblique – bas / droite

Langages : Python ou PHP (débutant)

En entrée : une valeur numérique donnée par l'utilisateur.

Le programme affiche une barre oblique qui commence en bas à droite.

```
Indiquer une valeur : 5
```

```
    *  
  *  
 *  
*  
*
```

# Série 6

Boucle for – condition if



## 6a - Croix en forme de x

Langages : Python ou PHP (débutant)

En entrée : une valeur numérique donnée par l'utilisateur.

Le programme affiche une croix en forme de x.

Indiquer une valeur : 5

```
*  *  
 * *  
  *  
 * *  
*  *
```

## 6b - Contour du losange

Langages : Python ou PHP (débutant)

En entrée : une valeur numérique donnée par l'utilisateur.

Le programme affiche le contour d'un losange.

Indiquer une valeur : 5

```
*  
* *  
*   *  
* *  
*
```

## 6c - Croix en forme de +

Langages : Python ou PHP (débutant)

En entrée : une valeur numérique donnée par l'utilisateur.

Le programme affiche une croix en forme de +.

```
Indiquer une valeur : 5
```

```
  *  
  *  
*****  
  *  
  *
```



## 6d - Triangle isocèle inverse vers le haut

Langages : le même que 4a

En entrée : une valeur numérique donnée par l'utilisateur.

Même consigne que 4a avec une boucle while.

Indiquer une valeur : 5

\*\*\*\* \*

\*\*\* \*\*

\*\* \*\*\*

\* \*\*\*\*

## 6e - Croix en forme de x

Langages : le même que 6a

En entrée : une valeur numérique donnée par l'utilisateur.

Même consigne que 6a avec une boucle while.

Indiquer une valeur : 5

```
*  *  
 * *  
  *  
 * *  
*  *
```

## 6f - Contour du losange

Langages : le même que 6b

En entrée : une valeur numérique donnée par l'utilisateur.

Même consigne que 6b avec une boucle while.

Indiquer une valeur : 5

```
*  
* *  
*   *  
* *  
*
```

# Série 7

Fonctions

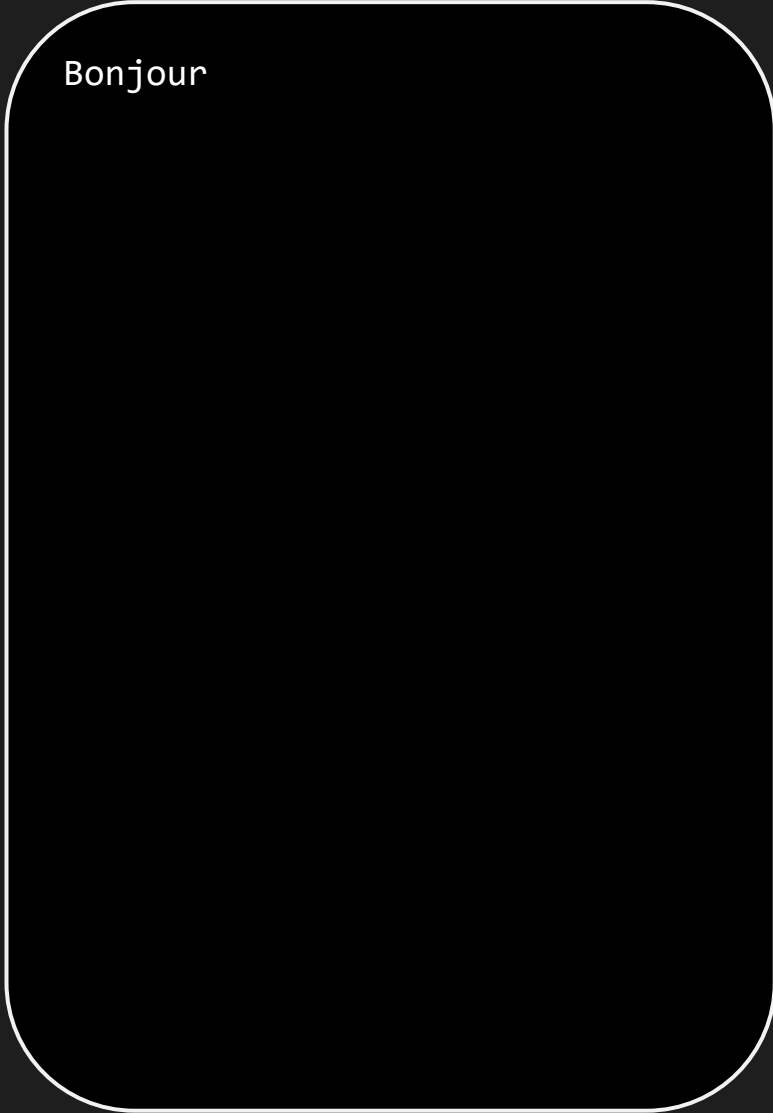


# 7a – Créer et appeler une fonction

Langages : Python ou PHP (débutant)

En entrée : rien

Le programme appelle la fonction `direBonjour()`.



Bonjour

# 7b – Créer et appeler une fonction avec un paramètre

Langages : Python ou PHP (débutant)

En entrée : un prénom saisi par l'utilisateur.

Le programme appelle la fonction `direBonjour()` avec la saisie de l'utilisateur.

```
Indiquer un prénom :  
Edgar  
Bonjour Edgar
```

# 7c – Créer et appeler une fonction avec un paramètre optionnel

Langages : Python ou PHP (débutant)

En entrée : un prénom saisi par l'utilisateur.

Le programme appelle la fonction `direBonjour()` avec la saisie de l'utilisateur (comme à 7b) puis appelle à nouveau la fonction sans paramètre.

```
Indiquer un prénom :  
Edgar  
Bonjour Edgar  
Bonjour Inconnu
```

# 7d – Créer et appeler une fonction avec plusieurs paramètres

Langages : Python ou PHP (débutant)

En entrée : un prénom puis un âge saisi par l'utilisateur.

Le programme appelle la fonction `direBonjour()` avec le prénom et l'âge saisis par l'utilisateur.

```
Indiquer un prénom :  
Edgar  
Indiquer un âge :  
18  
Bonjour Edgar, tu as 18 ans
```



# 7d – Créer et appeler une fonction avec plusieurs paramètres

Langages : Python ou PHP (débutant)

En entrée : un prénom puis un âge saisi par l'utilisateur.

Le programme appelle la fonction `direBonjour()` avec le prénom et l'âge saisis par l'utilisateur.

```
Indiquer un prénom :  
Edgar  
Indiquer un âge :  
18  
Bonjour Edgar, tu as 18 ans
```

## 7e – Créer et appeler une fonction avec plusieurs paramètres obligatoires et facultatifs

Langages : Python ou PHP (débutant)

En entrée : un prénom puis un âge saisi par l'utilisateur.

Le programme appelle la fonction `direBonjour()` avec le prénom et l'âge saisis par l'utilisateur puis une nouvelle fois avec uniquement le prénom.

Indiquer un prénom :

Edgar

Indiquer un âge :

18

Bonjour Edgar, tu as 18 ans

Bonjour Edgar

# 7f – Créer et appeler une fonction avec plusieurs paramètres facultatifs

Langages : Python ou PHP (débutant)

En entrée : un prénom puis un âge saisi par l'utilisateur.

Le programme appelle la fonction `direBonjour()` :

- Avec le prénom et l'âge saisis par l'utilisateur,
- Avec uniquement le prénom,
- Sans aucun paramètre.

Indiquer un prénom :

Edgar

Indiquer un âge :

18

Bonjour Edgar, tu as 18 ans

Bonjour Edgar

Bonjour Inconnu à l'âge  
indéterminé

# Série 8

Boucle foreach



# Prérequis de la série 8

Langages : PHP

En entrée : le tableau multi-dimensionnel ci-dessous :

```
$tableau = array("Citron", "Kiwi", array("Pomme", "Poire", "Fraise", array("Framboise", "Cerise"), array("Banane", "Pêche", array("Abricot"), "Nectarine"), "Orange", "Pamplemousse"), "Bergamote", array("Clémentine", "Mandarine"), "Groseille");
```

# 8a – Afficher les chaînes d'un tableau

Langages : PHP

En entrée : la variable \$tableau

Afficher les entrées de \$tableau qui ne sont pas elles-mêmes des tableaux.

```
Citron  
Kiwi  
Bergamote  
Groseille
```

## 8b – Afficher les valeurs et les indices d'un tableau

Langages : PHP

En entrée : la variable \$tableau

Afficher les indices et les entrées de \$tableau qui ne sont pas elles-mêmes des tableaux.

```
[0] : Citron  
[1] : Kiwi  
[3] : Bergamote  
[5] : Groseille
```

# 8c – Afficher les valeurs et les indices d'un tableau à deux dimensions

Langages : PHP

En entrée : la variable \$tableau

Afficher les indices et les entrées de \$tableau sur les 2 premières dimensions.

```
[0] : Citron  
[1] : Kiwi  
[0] : Pomme  
[1] : Poire  
[2] : Fraise  
[5] : Orange  
[6] : Pamplemousse  
[3] : Bergamote  
[0] : Clémentine  
[1] : Mandarine  
[5] : Groseille
```



# Série 9

Boucle foreach et fonctions



# Prérequis de la série 9

Langages : PHP

En entrée : le tableau multi-dimensionnel de la série 8

# 9a – Afficher les valeurs et les indices d'un tableau

Langages : PHP

En entrée : la variable \$tableau

Afficher les indices et les entrées de \$tableau qui ne sont pas elles-mêmes des tableaux en utilisant une fonction.

```
[0] : Citron  
[1] : Kiwi  
[3] : Bergamote  
[5] : Groseille
```

## 9b – Afficher un tableau et ses sous-tableaux

Langages : PHP

En entrée : la variable \$tableau

Afficher les indices et les entrées de \$tableau sur toutes ses dimensions avec une fonction.

```
[0] : Citron  
[1] : Kiwi  
[0] : Pomme  
[1] : Poire  
[2] : Fraise  
[0] : Framboise  
[1] : Cerise  
[0] : Banane  
[1] : Pêche  
[0] : Abricot  
[3] : Nectarine  
[5] : Orange  
[6] : Pamplemousse  
[3] : Bergamote  
[0] : Clémentine  
[1] : Mandarine  
[5] : Groseille
```

# 9c – Afficher un tableau et ses sous-tableaux

Langages : PHP

En entrée : la variable \$tableau

Afficher les indices et les entrées de \$tableau sur toutes ses dimensions en mettant en évidence la profondeur de l'élément.

```
[0] : Citron
[1] : Kiwi
      [0] : Pomme
      [1] : Poire
      [2] : Fraise
            [0] : Framboise
            [1] : Cerise
            [0] : Banane
            [1] : Pêche
                  [0] : Abricot
            [3] : Nectarine
      [5] : Orange
      [6] : Pamplemousse
[3] : Bergamote
      [0] : Clémentine
      [1] : Mandarine
[5] : Groseille
```

# 9d – Afficher un tableau et ses sous-tableaux

Langages : PHP

En entrée : la variable \$tableau

Afficher les indices et les entrées de \$tableau sur toutes ses dimensions avec les coordonnées de l'élément.

```
tableau[0] : Citron
tableau[1] : Kiwi
tableau[2][0] : Pomme
tableau[2][1] : Poire
tableau[2][2] : Fraise
tableau[2][3][0] : Framboise
tableau[2][3][1] : Cerise
tableau[2][4][0] : Banane
tableau[2][4][1] : Pêche
tableau[2][4][2][0] : Abricot
tableau[2][4][3] : Nectarine
tableau[2][5] : Orange
tableau[2][6] : Pamplemousse
tableau[3] : Bergamote
tableau[4][0] : Clémentine
tableau[4][1] : Mandarine
tableau[5] : Groseille
```

# Série 10

Boucle foreach et fonctions



# Prérequis de la série 10

Langages : PHP

En entrée : le tableau multi-dimensionnel ci-dessous :

```
$tableau = array("Kiwi", array("Pomme", "Poire", "Fraise", array("Framboise", "Cerise"), array("Citron", "Banane",  
"Pêche", array("Abricot", "Citron"), "Nectarine"), "Orange", "Pamplemousse", "Citron"), "Bergamote", "Citron",  
array("Clémentine", "Mandarine"), "Groseille");
```



# 10a – Rechercher la valeur « Citron »

Langages : PHP

En entrée : la variable \$tableau

Afficher les coordonnées de la valeur « Citron ».

```
$tableau[1][4][0] : Citron  
$tableau[1][4][3][1] : Citron  
$tableau[1][7] : Citron  
$tableau[3] : Citron
```

# 10b – Rechercher la première valeur « Citron »

Langages : PHP

En entrée : la variable \$tableau

Afficher les coordonnées de la première valeur « Citron ».

```
$tableau[1][4][0] : Citron
```

# 10c – Compter le nombre de « Citron »

Langages : PHP

En entrée : la variable \$tableau

Afficher le nombre de « Citron » trouvés dans le tableau.

4

# Série 11

Boucle foreach et fonctions



# Prérequis de la série 11

Langages : PHP

En entrée : le tableau multi-dimensionnel ci-dessous :

```
$tableau = array("Soda", "Soupe", "Pain", "Raclette", "Lasagnes", "Pizza", "Sandwich", "Hamburger", "Lait", "Salade",  
    "Riz", "Jambon", "Sucre", "Beurre", "Vanille");
```

# 11a – Afficher les mots ayant un nombre de lettres

Langages : PHP

En entrée : une valeur numérique donnée par l'utilisateur.

Afficher les mots qui ont le nombre de lettres indiqué par l'utilisateur.

Nombre de lettres : 5

Soupe

Pizza

# 11b – Afficher les mots ayant un nombre de lettres

Langages : PHP

En entrée : une valeur numérique donnée par l'utilisateur.

Afficher les mots qui ont le nombre de lettres indiqué par l'utilisateur, puis le nombre de mots qui ont été trouvés avec l'accord en nombre du texte.

```
Nombre de lettres : 6  
Salade  
Jambon  
Beurre  
3 trouvés
```

# 11c – Afficher les mots ayant un nombre de lettres

Langages : PHP

En entrée : une valeur numérique donnée par l'utilisateur.

Afficher le nombre de mots qui ont été trouvés puis les mots qui ont le nombre de lettres indiqué par l'utilisateur.

```
Nombre de lettres : 7  
1 trouvé  
Vanille
```



# 11d – Afficher les mots ayant un nombre de lettres

Langages : PHP

En entrée : une valeur numérique donnée par l'utilisateur.

Afficher le nombre de mots qui ont été trouvés puis les mots qui ont le nombre de lettres indiqué par l'utilisateur.

```
Nombre de lettres : 8  
3 mots trouvés de 8 lettres :  
Raclette  
Lasagnes  
Sandwich
```

# 11e – Afficher les mots ayant un nombre de lettres

Langages : PHP

En entrée : une valeur numérique donnée par l'utilisateur.

Refaire l'exercice 11d avec une fonction.

```
Nombre de lettres : 8  
3 mots trouvés de 8 lettres :  
Raclette  
Lasagnes  
Sandwich
```

# 11f – Afficher les mots ayant un nombre de lettres

Langages : PHP

En entrée : une valeur numérique donnée par l'utilisateur.

Afficher les mots ayant le bon nombre de lettres de 3 à 10 lettres.

```
1 mot trouvé de 3 lettres
Riz
3 mots trouvés de 4 lettres
Soda
Pain
Lait
2 mots trouvés de 5 lettres
Soupe
Pizza
3 mots trouvés de 6 lettres
Salade
Jambon
Beurre
1 mot trouvé de 7 lettres
Vanille
3 mots trouvés de 8 lettres
Raclette
Lasagnes
Sandwich
1 mot trouvé de 9 lettres
Hamburger
0 mot trouvé de 10 lettres
```

# 11f – Afficher les mots ayant un nombre de lettres

Langages : PHP

En entrée : une valeur numérique donnée par l'utilisateur.

Afficher les mots ayant le bon nombre de lettres de 3 à 10 lettres.

```
1 mot trouvé de 3 lettres
Riz
3 mots trouvés de 4 lettres
Soda
Pain
Lait
2 mots trouvés de 5 lettres
Soupe
Pizza
3 mots trouvés de 6 lettres
Salade
Jambon
Beurre
1 mot trouvé de 7 lettres
Vanille
3 mots trouvés de 8 lettres
Raclette
Lasagnes
Sandwich
1 mot trouvé de 9 lettres
Hamburger
0 mot trouvé de 10 lettres
```

# Masque de sous-réseau

Boucle for, switch-case



# Convertisseur de masque en notation CIDR en binaire

Langages : PHP, Python, Java

En entrée : une valeur de masque de sous-réseau en notation CIDR.

Afficher le masque en binaire avec séparation entre chaque octet.

```
Saisir un masque : 24  
11111111.11111111.11111111.0000  
0000
```

# Vérificateur de masque de sous-réseau

Langages : PHP, Python, Java

En entrée : un tableau avec des valeurs de masque de sous-réseau.

Afficher toutes les entrées du tableau et pour chaque valeur affiche si elle est correcte ou incorrecte.

```
255.128.0.0 est correct  
255.255.0.128 est incorrect  
255.193.0.0 est incorrect  
255.240.0.0 est correct  
255.255.192.0 est correct  
255.0.254.0 est incorrect  
255.0.0.0.0 est incorrect  
255.0.0 est incorrect
```

# Convertisseur de masque en notation décimale en CIDR

Langages : PHP, Python, Java

En entrée : des masques de sous-réseau en notation décimale.

Afficher le masque en notation CIDR si il est valide.

```
255.128.0.0 est égal à 9  
255.255.0.128 est incorrect  
255.193.0.0 est incorrect  
255.240.0.0 est égal à 12  
255.255.192.0 est égal à 18  
255.0.254.0 est incorrect  
255.0.0.0.0 est incorrect  
255.0.0 est incorrect  
255.255.255.252 est égal à 30
```



# Numerus Romanus

Écrire un nombre entier en chiffres romains



# Numerus Romanus - 1

Langages : PHP, Java, Python, JS

En entrée : un nombre entier saisi par l'utilisateur

En sortie : le nombre entier écrit en chiffres romains

Exemples à tester : voir ci-contre

```
7 => VII
13 => XIII
42 => XLII
79 => LXXIX
246 => CCXLVI
439 => CDXXXIX
777 => DCCLXXVII
908 => CMVIII
1998 => MCMXCVIII
2019 => MMXIX
3888 => MMMDCCCLXXXVIII
```

# Numerus Romanus - 2

Langages : PHP, Java, Python, JS

En entrée : un nombre entier écrit en chiffres romains saisi par l'utilisateur

En sortie : le nombre entier écrit en notation décimale

Exemples à tester : voir ci-contre

Tout caractère qui n'est pas utilisé par les chiffres romains doit déclencher un message d'erreur.

```
VII => 7  
XIV > 14  
XLII => 42  
LXXIX => 79  
CCXLVI => 246  
DCCXLIX => 749  
MDCXCV => 1695  
MMCDLXXIII => 2473  
MMMDCXCI => 3691  
MMMCMVII => 3907
```

# Nombres premiers

Identifier les nombres premiers



# Nombres premiers jusqu'à 14

Langages : PHP, Java, Python, JS

En sortie : le programme détermine pour tous les nombres de 1 à 14 (inclus) si ils sont :

- Premiers
- Ou pairs/impairs

```
1 est un nombre impair.  
2 est un nombre premier.  
3 est un nombre premier.  
4 est un nombre pair.  
5 est un nombre premier.  
6 est un nombre pair.  
7 est un nombre premier.  
8 est un nombre pair.  
9 est un nombre impair.  
10 est un nombre pair.  
11 est un nombre premier.  
12 est un nombre pair.  
13 est un nombre premier.  
14 est un nombre pair.
```

# Nombres premiers jusqu'à 48

Langages : PHP, Java, Python, JS

En sortie : le programme détermine pour tous les nombres de 1 à 48 (inclus) si ils sont :

- Premiers
- Ou pairs/impairs

```
1 est un nombre impair.  
2 est un nombre premier.  
3 est un nombre premier.  
4 est un nombre pair.  
5 est un nombre premier.  
6 est un nombre pair.  
7 est un nombre premier.  
8 est un nombre pair.  
9 est un nombre impair.  
10 est un nombre pair.  
11 est un nombre premier.  
12 est un nombre pair.  
13 est un nombre premier.  
14 est un nombre pair.
```

# Count

De Python3 à Java



# Créer la classe `Count` en Java

Langages : Python3 & Java

Le constructeur de la classe `Count` incrémente la variable statique `count` à chaque création d'une nouvelle instance.

La méthode d'instance `getCount()` retourne la valeur statique `Count.count` tant que `self.count` n'est pas initialisée.

```
2 2
8 49 3
```



# Version Python3

```
class Count:
    count = 0
    def __init__(self):
        Count.count += 1
    def getCount(self):
        return self.count
    def setCount(self, value):
        self.count = value

v1 = Count()
v2 = Count()
print(v1.getCount(), v2.getCount())
v1.setCount(8)
v2.setCount(49)
v3 = Count()
print(v1.getCount(), v2.getCount(), v3.getCount())
```

# Version Java, la méthode `main()`

```
public static void main(String[] args) {  
    Count v1 = new Count();  
    Count v2 = new Count();  
    System.out.println(v1.getCount() + " " + v2.getCount());  
    v1.setCount(8);  
    v2.setCount(49);  
    Count v3 = new Count();  
    System.out.println(v1.getCount() + " " + v2.getCount() + " " + v3.getCount());  
}
```

# PHPArray

De PHP à Java



# Créer la classe **PHPArray** en Java

Langages : PHP & Java

PHP est un langage à typage dynamique faible, il peut gérer des tableaux avec des valeurs d'index numériques ou alphanumériques.

- Les tableaux n'ont pas de taille fixe.
- L'index commence à 0.
- Si l'index n'est pas précisé, une nouvelle entrée sera à l'index numérique maximum précédent +1.
- La valeur peut être de n'importe quel type, dont un celui d'un autre tableau.

```
Index : 0, value = 12
Index : 1, value = Citron vert
Index : 2, value = 25
Index : 3, value = Orange
Index : 8, value = Pamplemousse
Index : a, value = Bergamote
Index : 9, value = Pomelos
Index : 10, value = Cédrat
Index : 11, value = 17
Index : agrume, value = Limette
Index : 12, value is an array
```

# Version PHP

```
<?php
$myArr = array(12, "Citron vert", 25);
$myArr[] = "Citron";
$myArr[8] = "Pamplemousse";
$myArr['a'] = "Bergamote";
$myArr[] = "Pomelos";
$myArr[3] = "Orange";
$myArr[] = "Cédrat";
$myArr[] = 17;
$myArr['agrume'] = "Limette";
$myArr[] = array("Mandarine", "Clémentine");
foreach($myArr as $index => $value) { /* affichage du tableau */ }
?>
```

# Version Java, la méthode `main()`

```
public static void main(String[] args) {  
    PHPArray myArr = new PHPArray(12, "Citron vert", 25);  
    myArr.add("Citron");  
    myArr.add("Pamplemousse", 8);  
    myArr.add("Bergamote", "a");  
    myArr.add("Pomelos");  
    myArr.add("Orange", 3);  
    myArr.add("Cédrat");  
    myArr.add(17);  
    myArr.add("Limette", "agrume");  
    myArr.add(new Object[] {"Mandarine", "Clémentine"});  
    for(Object item : myArr.toArray()) { /* affichage du tableau */ }  
}
```