
# Premiers pas en Algo - C

## Objectifs du module

L'objectif de ce module est l'apprentissage des concepts communs à tous les langages informatiques.

## Compétences

Renforcement des notions suivantes:

- paramètres et valeur de retour d'une fonction
- conditions et opérateurs logiques
- les branchements et les boucles

## Démarche pédagogique

**Tous les exercices doivent être réalisés seul(e), en autonomie. Les livrables doivent être validés par le formateur avant de passer à l'exercice suivant.**

# Exercice C.1 - 1, 2, 3, 4, …….

## Énoncé

Écrire un programme, en utilisant une boucle `while`, qui affiche tous les nombres de 1 à 100.

Il existe une autre boucle, plus simple à utiliser que la boucle `while` quand on veut parcourir une série de nombres, c'est la boucle `for` :

Ré-écrivez le programme précédent avec une boucle for.

## Ressources

- https://processing.org/reference/for.html
- https://www.tutorialspoint.com/java/java_for_loop.htm
- https://www.w3schools.com/java/java_for_loop.asp
- https://fr.flossmanuals.net/processing/les-repetitions/

## Compétences

- Programmer une boucle

# Exercice C. 2 - 2, 4, 6, 8, …….

Écrire un programme qui affiche tous les nombres pairs entre 1 et 100

# Exercice C.3 - 1, 2, 4, 8, 16, 32, …….

Écrire un programme qui affiche toutes les puissances de deux, jusqu'à 1024. C'est à dire que pour passer d'une valeur à la suivante, on multiplie par 2.

# Exercice C.4 - 1, 4, 9, 16, 25, …….

Écrire un programme qui affiche le carré, c'est à dire n*n de chaque nombre entre 1 et 100.

## Exercice C.5 - Plus petit carré supérieur 🙄

En vous inspirant de l'exercice 4, écrivez une fonction qui prend un nombre entier en paramètre et retourne le plus petit carré supérieur à n. Par exemple: pour n=30 -> 36, et pour n=63 -> 64

## Exercice C.6 - Plus grand carré inférieur 😯

En vous inspirant de l'exercice précédent, écrivez une fonction qui prend un nombre entier en paramètre et retourne le plus grand carré inférieur à n. Par exemple: pour n=30 -> 25, et pour n=63 -> 49.

## Exercice C.7 - Somme de 1 à 100

### Énoncé

Écrire une fonction qui calcule la somme de tous les nombres de 1 à 100 c'est à dire 1 + 2 + 3 + ….. + 99 + 100

Pour cela vous aurez besoin:

- d'une variable pour parcourir les nombres de 1 à 100 (comme de l'exo 1)
- d'une variable pour calculer cette somme au fur et à mesure que vous parcourez tous les nombres.

### Livrables

- Une fonction qui calcule la somme de 1 à 100

### Note

En cas de difficulté, et c'est normal, essayez de faire votre algo avec deux post-its, un pour chaque variable. Quelles opérations faut-il faire sur ces variables à chaque tour de boucle pour calculer la somme de 1 à 6 par exemple ?

### Bonus

Modifiez la fonction pour qu'elle calcule la somme des nombres de 1 à `n` ou `n` est un paramètre de la fonction.

# Exercice C.8 - Suite de Syracuse

## Enoncé

En mathématiques, on appelle suite de Syracuse une suite d'entiers naturels définie de la manière suivante : on part d'un nombre entier plus grand que zéro ; s'il est pair, on le divise par 2 ; s'il est impair, on le multiplie par 3 et on ajoute 1. En répétant l'opération, on obtient une suite d'entiers positifs dont chacun ne dépend que de son prédécesseur.

Par exemple, à partir de 14, on construit la suite des nombres : 14, 7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1, 4, 2, 1, 4, 2… C'est ce qu'on appelle la suite de Syracuse du nombre 14.

Écrire une fonction qui prend en paramètre un entier n et affiche la suite de syracuse correspondante (en s'arrêtant quand la valeur 1 est atteinte).

## Note

Pour savoir si un nombre est pair, on teste si le reste de la division entière par 2 est égale à 0. Une division entière, c'est de partager en parts égales quelque chose que l'on ne peut pas couper, par exemple des billes. Si on essaie de partager 15 billes entre deux enfants, ils auront chacun 7 billes et il restera une bille.

Il existe un opérateur qui calcule directement le reste de la division entière, il s'appelle modulo et en java, s'écrit `%`.

Par exemple:
15 % 2 vaut 1 ( car si on divise 15 par 2, on obtient 7 et il reste 1)
15 % 3 vaut 0 ( car si on divise 15 par 3, on obtient 5 et il reste 0)
15 % 4 vaut 3 ( car si on divise 15 par 4 on obtient 3 et il reste 3)

# Exercice Bonus

## Énoncé
Écrire une fonction qui indique si un nombre est premier.

## Note
Un nombre premier n'est divisible que par 1 et par lui-même. C'est à dire qu'il n'existe aucun nombre (autre que 1 et lui-même)  pour lequel le reste de la division entière est nul.
Pour savoir si un nombre est premier, il faut donc tester le modulo de ce nombre (voir exercice précédent) avec tous les nombres qui sont plus petits que lui (en fait on peut se contenter de vérifier tous les nombres plus petits que la racine carrée de ce nombre). Si tous les modulos sont différents de 0, alors ce nombre est premier.

## Livrables
- Une fonction isPrime qui prend un entier en paramètre et renvoie un booléen indiquant si ce nombre est premier

# Exercice Bonus 2

## Énoncé
En utilisant la fonction précédente, afficher tous les nombre premiers de 1 à 1000