Exercices

Exercice 1 * Écrire la fonction **est_pair(x : int) → bool** qui renvoie **True** si l'entier **x** est pair, **False** sinon.

Exercice 2 * Écrire la fonction **valeur_absolue(x: int) → int** qui renvoie la valeur absolue de l'entier **x**.

Exercice 3 * Écrire la fonction **surface(r: float) → float** qui renvoie l'aire d'un cercle de rayon **r**.

Exercice 4 * Écrire une fonction **volume(longueur: int, largeur: int, hauteur: int) → int** qui calcule le volume d'un pavé.

Exercice 5 * Écrire la fonction **est_majeur(age: int) → bool** qui renvoie **True** si la personne d'âge age est majeure, **False** sinon.

Exercice 6 * Écrire la fonction **puissance(x: int, n: int) → int** qui renvoie **x** à la puissance **n**. Pour rappel $x^n = \underbrace{x \times x \times \cdots \times x}_{n}$. On utilisera une boucle pour effectuer le calcul.

Exercice 7 * Écrire la fonction **lancer_des() → int** qui simule le lancer de deux dés à six faces et renvoie la somme des valeurs obtenues.

Exercice 8 * Écrire la fonction **pythagore(a: int, b: int, c: int) → bool** qui renvoie **True** si le triangle formé par les côtés de mesures **a b c** est rectangle. On supposera que les mesures sont des entiers donnés dans l'ordre croissant.

Exercice 9 ** Écrire la fonction **somme(n: int) → int** qui renvoie la somme des entiers de 1 à **n**.

Exercice 10 ** Un entier naturel est dit parfait lorsqu'il est égal à la somme de tous ses diviseurs propres autres que lui-même. Par exemple 6 est parfait car $6 = 1 + 2 + 3$. Écrire la fonction **est_parfait(n: int) → bool** qui vérifie si **n** est un nombre parfait. **Indication:** Un nombre **i** est un diviseur de **n** si le reste de la division de **n** par **i** est nul.

Exercice 11 *** Écrire la fonction **est_premier(n: int) → bool** qui renvoie **True** si **n** est premier. Pour rappel, un nombre est premier s'il n'est divisible que par 1 et lui-même.

Exercice 12 *** La conjecture de COLLATZ est un énoncé mathématique dont on ne sait pas encore s'il est vrai. Prendre un nombre entier positif, et lui appliquer lui le traitement suivant :

- s'il est pair, le diviser par 2
- sinon, le multiplier par 3 et ajouter 1

On obtient alors un nouveau nombre, sur lequel la procédure est répétée.

La conjecture s'énonce ainsi : **quel que soit l'entier choisi au départ, on finira par obtenir 1.**

Écrire une fonction **collatz(n: int) → bool** qui renvoie **True** si le processus s'achève, c'est à dire si on finit par atteindre 1.

Exercice 13 ***

1. Écrire une fonction **triangle(c: int) → None** qui trace un triangle de côté **c**.

2. Écrire le programme qui appelle la fonction et affiche la figure: 

Exercice 14 *

1. Créer un fichier **mes_fonctions.py**
2. Dans ce fichier, copier les fonctions de l'exercice 1, 2, 4, 5, 6
3. Télécharger l'annexe **tests** et extraire le fichier **tests_unitaires.py** dans le même répertoire que **mes_fonctions.py**
4. Lancer le fichier **tests_unitaires.py**