Exercice 1.
On considère la fonction factorielle $n!$ définie par $n! = 1 \times 2 \times \cdots \times n$ si $n > 0$ et $0! = 1$.

a. Donner une définition récursive de la fonction mathématiques $factorielle(n)$.

b. Ecrire une fonction Python `fact(n)` qui implémente cette définition.

c. Dessiner l'arbre d'appels correspondant au calcul de `fact(4)`.

Exercice 2.
La méthode du paysan russe est un très vieil algorithme de multiplication de deux nombres entiers déjà décrit sur un papyrus égyptien rédigé autour de 1650 avant J.-C. Il s'agissait de la principale méthode de multiplication en Europe avant l'introduction des chiffres arabes.
Cet algorithme repose sur les relations :

$$x \times y = \begin{cases} 0 & \text{si } x = 0 \\ (x//2) \times (2y) & \text{si } x \text{ est pair} \\ (x//2) \times (2y) + y & \text{si } x \text{ est impair} \end{cases}$$

Ecrire une fonction récursive `multiplication_russe(x,y)` qui calcul le produit de `x` et `y` en utlisant la méthode du paysan russe.

Exercice 3.
On considère la suite u_n définie par la relation de récurrence suivante, où a et b sont des réels quelconques :

$$u_n = \begin{cases} a & \text{si } n = 0 \\ b & \text{si } n = 1 \\ 3u_{n-1} + 2u_{n-2} + 5 & \text{si } n \geq 2 \end{cases}$$

Ecrire une fonction `suite(n, a, b)` qui renvoie le `n`-ème terme de cette suite pour des valeurs `a` et `b` données en paramètres.

Exercice 4.
Ecrire une fonction récursive `boucle(i,k)` qui affiche les entiers entre `i` et `k`. Par exemple, `boucle(1,4)` doit afficher `1 2 3 4`.

Exercice 5.
Ecrire une fonction récursive `nombre_de_chiffres(n)` qui prend un entier positif ou nul `n` en argument et renvoie son nombre de chiffres. Par exemple, `nombre_de_chiffres(12345)` doit renvoyer 5.

Exercice 6.
En vous inspirant de l'exercice précédent, écrire une fonction récursive `nombre_de_bits_1(n)` qui prend un entier positif ou nul et renvoie le nombre de bits valant 1 dans la représentation binaire de `n`. Par exemple `nombre_de_bits_1(255)` doit renvoyer 8.

Remarque : cette fonction est utile dans de nombreux algorithmes bas niveau et porte le nom de `popcount` dans la litérature (de l'anglais *population count*). Elle est souvent implée dans les unité arithmétiques et logiques des processeurs sur des entiers de taille fixe (32 ou 64 bits).

Exercice 7.
Écrire une fonction récursive `appartient(v, t, i)` prenant en paramètres une valeur `v`, un tableau `t` et un entier `i` et renvoyant `True` si `v` apparaît dans `t` entre l'indice `i` (inclus) et `len(t)` (exclu), et `False` sinon. Dans un premier temps on supposera que `i` est toujours compris entre `i` et `len(t)`, puis on utilisera l'instruction `assert` pour restreindre les appels à la fonction `appartient(v, t, i)`.

Exercice 8.
La courbe de Koch est une fractale imaginé par le mathématicien suédois Niels Fabian Helge von Koch en 1904.
On peut la créer à partir d'un segment de droite, en modifiant récursivement chaque segment de droite de la façon suivante :

— On divise le segment de droite en trois segments de longueurs égales.
— On construit un triangle équilatéral sans base au-dessus du morceau central.

On répète ce processus n fois, n est appelé l'ordre.

n=0 n=1 n=2 n=3

Ecrire une fonction `koch(n,l)` qui dessine avec `Turtle` une courbe de Koch d'ordre `n` à partir d'un segment de longueur `l`.