Exercice 1 : La somme des entiers s'écrit :

$$0 + 1 + 2 + ... + n$$

1. Donner une définition récursive de la somme des entiers.
2. Implémenter la fonction **somme(n : int) → int**.

Exercice 2 : La fonction factorielle est définie par :

$$n! = 1{\times}2{\times}3...{\times}n \qquad si \qquad n > 0 \qquad et \qquad 0! = 1$$

1. Donner une définition récursive qui correspond au calcul de la fonction factorielle.
2. Implémenter la fonction **factorielle(n : int) → int**.

Exercice 3 : Soit u_n la suite d'entiers définie par $u_0 > 1$ et :

$$u_{n+1} = \begin{cases} u_n/2 & \text{si } u_n \text{ est pair} \\ 3{\times}u_n + 1 & \text{sinon} \end{cases}$$

Écrire la fonction *syracuse(u:int)* → *None* qui affiche les valeurs successives de la suite u_n tant que $u_n > 1$. L'appel de la fonction s'effectuera avec une valeur de u_0 quelconque.

Exercice 4 :

1. Écrire une fonction récursive **entiers(i : int, k : int) → None** qui affiche les entiers entre i et k. Par exemple, entiers(0,3) doit afficher 0 1 2 3.
2. Écrire une fonction récursive **impairs(i : int, k : int) → None** qui affiche les nombres impairs entre i et k.

Exercice 5 : Écrire la fonction récursive **pgcd(a : int, b : int) → int** qui renvoie le Plus Grand Commun Diviseur de a et b. On donne comme précondition : a < b.

méthode d'Euclide : (20,35)

$35 = \overbrace{20}^{a \to b} \times 1 + \overbrace{15}^{b\%a \to a}$
$20 = 15{\times}1 + 5$
$15 = 5{\times}3 + 0$
pgcd = 5

Exercice 6 : Écrire une fonction récursive **nombre_chiffres(n : int) → int** qui renvoie le nombre de chiffres qui compose n.

Exercice 7 : La formulation récursive ci-après permet de calculer les coefficients binomiaux :

$$C(n,p) = \begin{cases} 1 & \text{si } p = 0 \text{ ou } n = p \\ C(n-1,p-1) + C(n-1,p) & \text{sinon} \end{cases}$$

1. Écrire une fonction récursive **C(n : int, p : int) → int** qui renvoie la valeur de C(n,p).
2. Le triangle de Pascal est une présentation des coefficients binomiaux sous la forme d'un triangle. Dessiner le triangle de Pascal à l'aide d'une double boucle *for* pour n variant de 0 à 10.

Christophe Viroulaud

Pour ceux qui ont fini : palindrome

```
def palindrome(s):
    if len(s)<2:
        return True
    elif not(s[0]==s[-1]):
        return False
    else:
        return palindrome(s[1:-1])

print(palindrome("ressasser"))
print(palindrome("ressadsser"))
```

tour de hanoi pour les plus avancés

Christophe Viroulaud