Chap. 1 — Récursivité

1.1 — Problème de la somme des n premiers entiers

Pour définir la somme des n premiers entiers, on utilise généralement la formule $0 + 1 + 2 + \ldots + n$. Cette formule parait simple mais elle n'est pas évidente à programmer en python.

Écrire une fonction `somme(n)` qui renvoie la somme des `n` premiers entiers.

CORRECTION

[1]:
```python
def somme(n):
    """
    Calcule la somme des n premiers entiers.
    param : n (int), dernier entier à ajouter

    exemples:
    >>> somme (0)
    0
    >>> somme (5)
    15
    """
    r = 0
    for i in range(n+1):
        r = r + i
    return r
```

On remarque que le code python n'a rien à voir avec sa formulation mathématique.

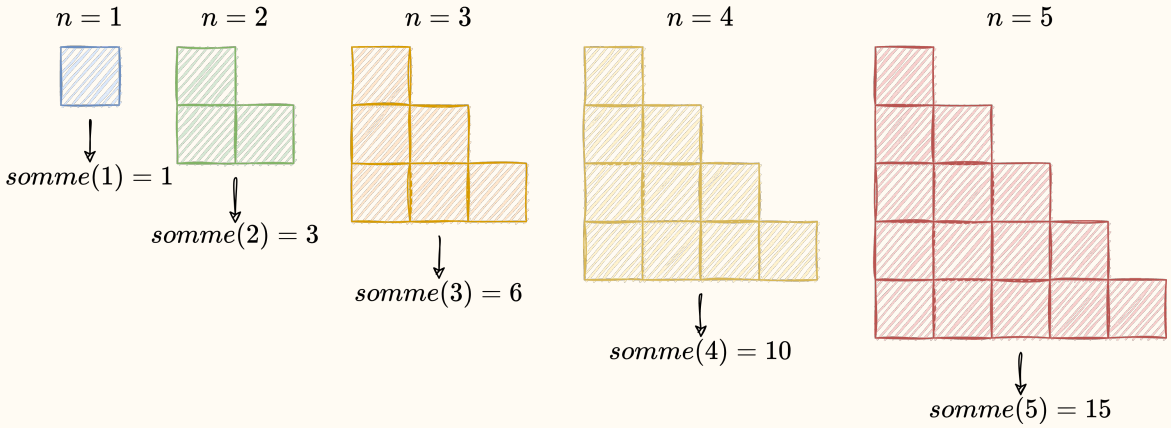
Nouvelle formulation

Il existe une autre manière d'aborder ce problème en définissant une fonction mathématique $somme(n)$.

ACTIVITÉ

1. **Calculer** $somme(0)$?

Utilisons maintenant cette illustration pour modéliser quelques exemples de calculs.



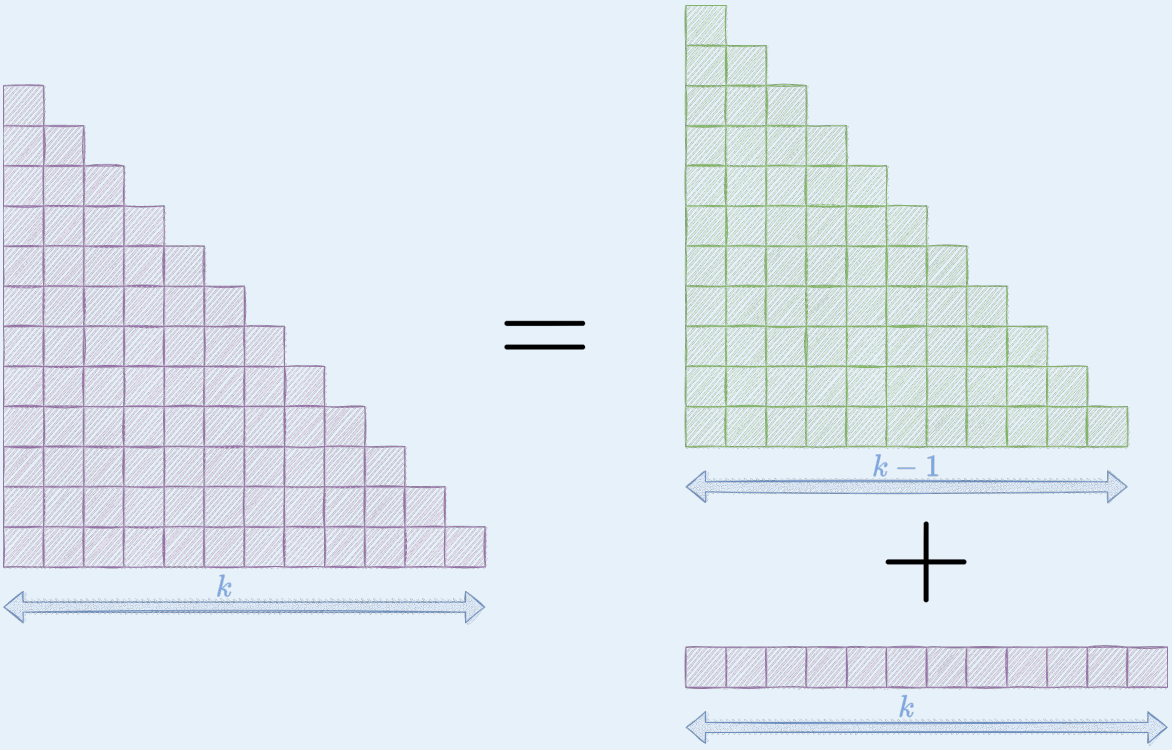
2. En observant ces exemples, **trouver une relation** entre :

- $somme(5)$ et $somme(4)$,
- $somme(4)$ et $somme(3)$.

3. Généraliser la relation entre $somme(n)$ et $somme(n-1)$.

CORRECTION

1. $somme(0) = 0$
2. On obtient :
   - $somme(5) = somme(4) + 5$
   - $somme(4) = somme(3) + 4$
3. En s'aidant du schéma



on obtient donc :

$$somme(n) = \begin{cases} 0 & \text{si } n = 0 \\ somme(n-1) + n & \text{si } n > 0 \end{cases}$$

Comme on peut le voir, la définition de $somme(n)$ dépend de la valeur de $somme(n-1)$.

Il s'agit d'une définition **récursive**, c'est-à-dire d'une définition de fonction qui fait appel à elle-même.

L'intérêt de cette définition récursive de la fonction $somme(n)$ est qu'elle est directement *calculable*, c'est-à-dire exécutable par un ordinateur.

ACTIVITÉ

En appliquant exactement la définition récursive de la fonction $somme(n)$, **programmer** une fonction `somme(n)` qui calcule la somme des `n` premiers entiers.

CORRECTION

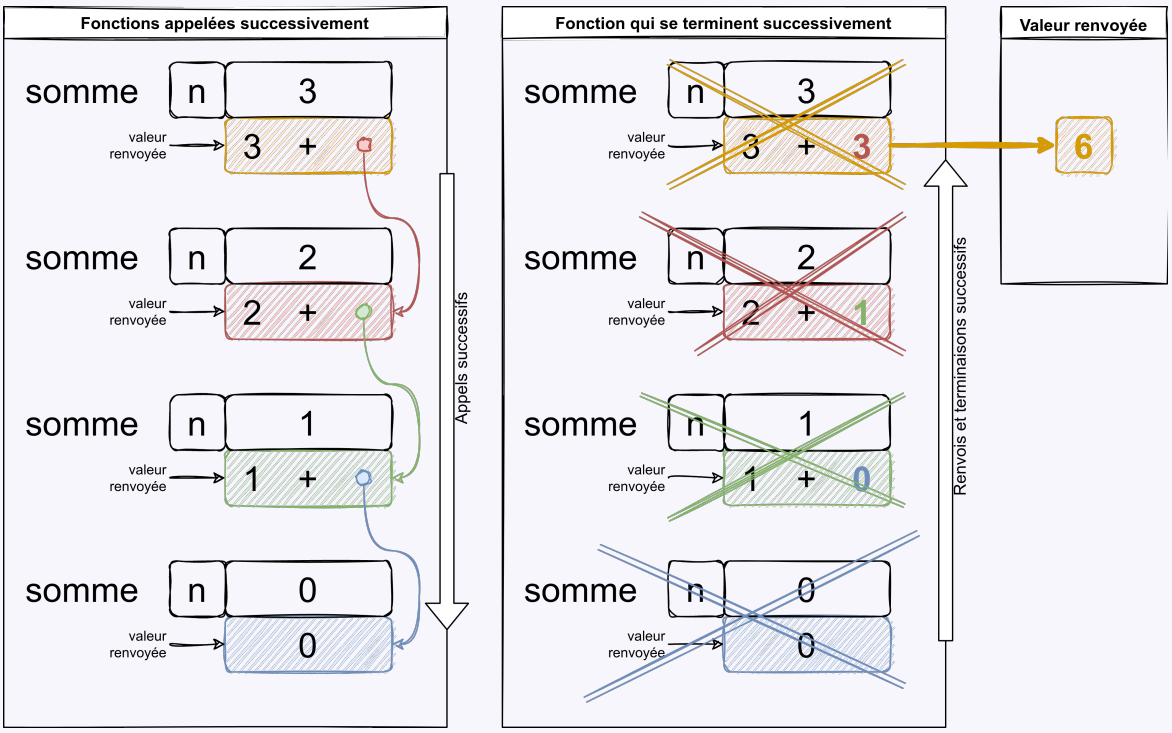
[2]:
```
# programmation défensive
import doctest
doctest.testmod()

def somme(n):
    """
    Calcule la somme des n premiers entiers.
    params: n (int), dernier entier à ajouter

    exemples:
    >>> somme (0)
    0
    >>> somme(10)
    55
    """
    if n==0:
        return 0
    else:
        return n + somme(n-1)
```

Exemple

Voici par exemple comment on peut représenter l'évaluation de l'appel à `somme(3)`



Pour calculer la valeur renvoyée par `somme(3)`, il faut d'abord appeler `somme(2)`. Cet appel va lui même déclencher un appel à `somme(1)`, qui a son tour nécessite un appel à `somme(0)`.

Ce dernier se termine directement en renvoyant la valeur 0. `somme(1)` peut alors se terminer et renvoyer le résultat de`1+0`. Enfin, l'appel à `somme(2)` peut lui même se terminer et renvoyer la valeur `2+1`.

Ce qui permet à `somme(3)` de se terminer en renvoyant le résultat `3+3`.

Ainsi on obtient bien la valeur 6 attendue!