Introduction

1 Dans la console

```
>>> 2**8
...

>>> 11/4
...

>>> 11//4
...

>>> type(11/4)
...

>>> type(11//4)
...
```

Les opérations % et //

```
>>> 11//4
...

>>> 11%4
...

>>> (11//4)*4 + (11%4)
...

>>> 11%2 == 1
...
```

Le range de Python!

```
>>> list(range(5))
...

>>> [k for k in range(5)]
...

>>> [k**2 for k in range(5)]
...

>>> [k**2 for k in range(5) if k%2 == 1]
...
```

Les booléens

Un booléen est un type de variables à deux états. Les booléens sont soit à l'état vrai, soit à l'état faux. En Python, les deux valeurs booléennes sont `True` et `False`.

```
>>> 2 < 3
True
>>> 2 == 3
False
>>> 2 != 3
True
>>> 2 <= 3
True
```

Il existe bien sûr des opérateurs logiques entre les booléens. On trouve `and`, `or`, `not`.
Vérifier que la table de vérité du « `or` » chez Python est la même que dans le cours de maths!

```
>>> True or False
...

>>> True or True
...

etc...
```

2 Ma première fonction : avec une boucle for

```
def MaPremiereFonction(n):
    S = 0
    for k in range(n+1):
        S = S + k**2
    return S
```

À la lecture, à votre avis, quel est le type de l'argument `n` ?
Sans utiliser l'ordinateur, que renvoie cette fonction ?

Pour la culture. Dans la console, en utilisant la fonction `sum` de Python, savez-vous comment faire le même travail en une seule ligne!
En général, on vous interdira d'utiliser les fonctions toutes faites de Python.

Toujours pour la culture. Savez-vous à quoi sert l'avant-dernière ligne dans la fonction ci-dessous ?

```
def MaPremiereFonctionBis(n):
    S = 0
    for k in range(n+1):
        S = S + k**2
    assert S == n*(n+1)*(2*n+1)//6
    return S
```

3 Boucle while

```
def toto(n):
    while n >= 1:
        print("pcsi 3, les stars !")
        n = n//2
```

Si l'on tape dans la console ce qui suit, que voit-on s'afficher ? Sans ordinateur.

```
>>> toto(7)
```

4 Ma fonction triple!

Écrire une fonction `MaFonctionTriple(a, n)` qui renvoie une liste contenant n nombres dont chaque terme est égal au triple du précédent, le terme initial étant a.
Règle du jeu. Évidemment, on n'utilisera pas la formule $a+3n$.
Par exemple, on doit avoir

```
>>> MaFonctionTriple(1, 10)
[1, 3, 9, 27, 81, 243, 729, 2187, 6561, 19683]
```

Une fois votre fonction codée, appelez-moi pour un défi.

Introduction

corrigés

4

```
71 def MaFonctionTriple(a,n):
72     L = []
73     for _ in range(n):
74         L.append(a)
75         a = 3*a
76     return L
```