Fonctions avec valeur de retour

1 Définition et appel

Il est possible pour une fonction de *retourner* une valeur, c'est-à-dire de transmettre une valeur au programme appelant qui pourra l'utiliser après appel de la fonction. Retourner une valeur se fait à l'aide de `return` suivi de la valeur à retourner.

```
def nom_fonction(parametre1, parametre2, ..., parametrek) :
    # algorithme de la fonction

    return resultat_a_retourner

# exemple d'appel où la valeur retournée est stockée dans la variable res
res = nom_fonction(valeur1, valeur2, ..., valeurk)
```

Remarques :

- Lors de l'appel d'une fonction, après son exécution, l'appel est remplacé dans le code du programme appelant par la valeur retournée par la fonction.
- L'instruction `return` termine l'exécution du code de la fonction. Elle peut être utilisée sans être suivie d'une valeur (juste pour stopper l'exécution).
- Une fonction ne retourne qu'une valeur. Si l'on souhaite retourner plusieurs valeurs, on retourne un tableau (chapitre suivant) ou un tuple (non abordé dans ce cours).

```
# définition d'une fonction calculant et retournant le périmètre d'un rectangle
def perimetre_rectangle(L, l):
    """
    Retourne le périmètre d'un rectangle calculé en fonction de L et l.
    L et l (entrée) : longueur et largeur du rectangle, de type float
    """
    return 2 * (L + l)  # le résultat est transmis au programme appelant

pr = perimetre_rectangle(5.2, 3.4) # pr contient 17.2 après appel de la fonction
print("Le périmètre du rectangle est : " + str(pr))
```

2 Tests unitaires

Un code contient généralement des bugs et les développeurs passent beaucoup de temps à tester/débugger leur code. Comme le nombre d'erreurs augmente avec la complexité d'un code, il est important de tester correctement chaque fonction. C'est ce que l'on appelle les *tests unitaires*.

Pour créer des tests unitaires :

- on crée une fonction de test pour chaque fonction que l'on définit. Par convention, le nom de la fonction de test d'une fonction `nomFonction` est `test_nomFonction`. La fonction de test ne prend aucun paramètre et ne retourne aucune valeur.
- Les tests consistent à appeler la fonction avec des valeurs d'entrée pour lesquelles on connaît le résultat
- Les tests entre la valeur attendue et la valeur retournée se font avec l'instruction `assert`.
- La fonction de test se termine par l'affichage `Test de la fonction ... : ok`.
- la fonction de tests est définie et appelée dans une même cellule du notebook.

```python
import math

# Fonction de tests unitaires de la fonction perimetre_rectangle
def test_perimetre_rectangle():
    assert perimetre_rectangle(5, 2) == 14
    assert perimetre_rectangle(0, 0) == 0
    assert math.isclose(perimetre_rectangle(0.2, 0.1), 0.6)
    print("Test de la fonction perimetre_rectangle : ok")

test_perimetre_rectangle()
```

Ainsi, à chaque fois que l'on modifie la fonction `perimetre_rectangle`, on appelle la fonction `test_perimetre_rectangle` pour vérifier que `perimetre_rectangle` vérifie les tests unitaires.

Important : Tester des fonctions utilisant des saisies clavier ou des affichages écran est techniquement un peu complexe et ne sera pas vu dans ce cours. Pour tester ces fonctions, on écrira alors en commentaire la liste des saisies clavier à tester et/ou la liste des appels de fonction à tester en indiquant quels messages doivent être affichés à l'écran.

2.1 Conception d'une fonction

Étape 1 : *déterminer le rôle de la fonction.*

- Il faut savoir ce que fait exactement la fonction, dans quel cadre elle sera utilisée, etc.
- Il faut trouver un nom de fonction explicite.

Étape 2 : *déterminer les paramètres et la valeur de retour (avec leur type).*

- Il faut écrire un exemple d'appel.
- Il faut écrire l'en-tête et la docstring.

Étape 3 : *écrire le corps de la fonction.*
Étape 4 : *écrire la fonction de tests unitaires.*

2.2 Vocabulaire

Valeur de retour, tests unitaires