Séance 2

LES FONCTIONS

L'essentiel

- Comme en mathématiques, une fonction reçoit des **arguments** et renvoie un résultat.
- On utilise le mot **`def`** pour définir une fonction. Lors de la déclaration d'une fonction, les valeurs d'entrées sont appelées **paramètres** et on indique ce qui est renvoyé par **`return`**.
- Par exemple, on peut définir la fonction `aire_carre` qui reçoit un paramètre `c` et qui renvoie la valeur de l'aire d'un carré de côté `c` :

```
def aire_carre(c):
        return c ** 2
```

- L'ensemble de ce qui doit être exécuté dans la fonction est **indenté**, c'est-à-dire décalé vers la droite avec la touche TAB :

```
def retourne(nombre):
    u = nombre % 10              # unités
    c = nombre // 100            # centaines
    d = (nombre - c*100) // 10   # dizaines
    return u*100 + d*10 + c
```

- Une fois la fonction déclarée et exécutée, on peut l'utiliser dans le script ou dans la console comme n'importe quelle fonction :

```
>>> retourne(123)
321
```

- Au premier `return` rencontré, la fonction renvoie la valeur demandée et quitte la fonction : la suite du code n'est pas exécutée. Dans l'exemple ci-dessous, la fonction renvoie 1 quelle que soit la valeur de `x` entrée :

```
def f(x):
        return 1
        return 3*x+1
```

Exercices du jour

Exercice 1 : Écrire une fonction `age` qui reçoit une année de naissance sur 4 chiffres et renvoie l'âge de la personne à ce jour (on ne tient compte que de l'année de naissance).

Exercice 2 : Statistiquement, la taille d'un enfant peut être estimée à partir de celles de ses parents.

- Pour un garçon, la formule permettant d'estimer sa taille à l'âge adulte est :

$$\frac{\text{taille de la mère en cm} + \text{taille du père en cm} + 13}{2}$$

- Pour une fille :

$$\frac{\text{taille de la mère en cm} + \text{taille du père en cm} - 13}{2}$$

1. Créer une fonction `fille(p, m)` qui reçoit la taille du père et de la mère, et qui renvoie la taille moyenne de leur fille à l'âge adulte.
2. Créer une fonction `fils(p, m)` sur le même principe pour un fils.
3. Utiliser ces 2 fonctions pour en créer une troisième : `enfant(p, m)` qui possède 2 paramètres : la taille du père, la taille de la mère et qui **affiche** : *"Si c'est un garçon, il mesurera ... cm et si c'est une fille elle mesurera ... cm, soit une moyenne de ... cm"*.

4. ★ En utilisant les fonctions `fille` et `fils` créer une fonction : `bonus(p, m, s)` qui possède 3 paramètres : la taille du père, la taille de la mère, et un dernier paramètre étant un entier indiquant le sexe de l'enfant (0 pour une fille et 1 pour un garçon) et renvoie la taille estimée pour l'enfant en fonction des paramètres d'entrée.

Exercice 3 : Un gardien de phare va aux toilettes cinq fois par jour. Or les WC sont au rez-de-chaussée...
Écrire une fonction `hauteurParcourue` qui reçoit deux paramètres, le nombre de marches du phare et la hauteur de chaque marche (en cm), et qui ne renvoie rien, mais **affiche** : *"Pour ... marches de ... cm, il parcourt ... m par semaine."*

Exercices en autonomie

Exercice 4 : Lorsque 2 résistances R_1 et R_2 sont montées en dérivation, on peut calculer la résistance équivalente grâce à l'égalité : $\frac{1}{R} = \frac{1}{R_1} + \frac{1}{R_2}$. Écrire une fonction qui reçoit deux nombres $R1$ et $R2$ et renvoie la valeur de la résistance équivalente.

Exercice 5 : On considère la fonction :

```
def arrondi(x) :
    return round(x) - round(x+1)
```

1. Que renvoient les appels `arrondi(0.3)`, `arrondi(1.9)`, `arrondi(-3.1)` et `arrondi(-3.9)` ?
2. Que peut-on conjecturer pour la valeur de `arrondi(x)` où `x` est un nombre quelconque ?
3. Que renvoient les appels `arrondi(0.5)` et `arrondi(1.5)` ?

Vous êtes surpris ? Exécuter étape par étape les calculs réalisés par Python pour tenter de comprendre... sinon rendez-vous à la partie **Solutions des exercices**.

Aide pour les exercices

Solutions des exercices

Correction 4 Sans calculs :

```
def Requiv(R1, R2):
    inverseR = 1/R1 + 1/R2
    return 1 / inverseR
```

ou plus court :

```
def Requiv(R1, R2):
    return 1 / (1/R1 + 1/R2)
```

Ou encore en remarquant que $\frac{1}{R} = \frac{1}{R_1} + \frac{1}{R_2} \Longleftrightarrow R = \frac{R_1 R_2}{R_1 + R_2}$:

```
def Requiv(R1, R2):
    return (R1*R2) / (R1 + R2)
```

Correction 5 Il suffit de taper le code demandé :

1. `arrondi(0.3)` = `arrondi(1.9)` = -1.
   De même `arrondi(-3.1)` = `arrondi(-3.9)` = -1.
2. On peut légitimement conjecturer que $\forall$ x $\in \mathbb{R}$, `arrondi(x)` = -1.
3. `arrondi(0.5)` = -2 et `arrondi(1.5)` = 0 !

Pour comprendre ce comportement surprenant, regardons ce que renvoie la fonction `round` sur quelques demi-entiers :

`round(-0.5)` $\mapsto$ 0	`round(0.5)` $\mapsto$ 0
`round(1.5)` $\mapsto$ 2	`round(2.5)` $\mapsto$ 2
`round(3.5)` $\mapsto$ 4	`round(4.5)` $\mapsto$ 4
`round(5.5)` $\mapsto$ 6	`round(6.5)` $\mapsto$ 6
`round(7.5)` $\mapsto$ 8	`round(8.5)` $\mapsto$ 8

L'arrondi de 3,5 est bien 4, mais l'arrondi de 4,5 est 4 aussi ! alors qu'avec la définition mathématique, il devrait être de 5.... et cela est dû.... à la finance ! Si on arrondit toujours avec la définition mathématique, tous les montants se terminant à 0,5 sont arrondis à l'unité supérieure entrainant ainsi des cumuls d'erreurs importants à la fin. Avec cette méthode les montants sont arrondis parfois au dessus, parfois en dessous, limitant ainsi les effets de seuil. En résumé, si le chiffre avant le 5 est impair on arrondit à l'entier supérieur et s'il est pair on arrondit à l'entier inférieur et inversement pour les négatifs ! Et c'est pareil pour les autres chiffres par exemple `round(2.65,1)=2.6` et `round(2.75,1) = 2.8`