I Fonctions

Une *fonction* est un script qui commence par le mot **def**.
On indique ensuite le nom de la fonction. Elle reçoit en entrée zéro, une ou plusieurs variables d'entrée (entre parenthèses après le nom de la fonction).
On termine cette ligne par **:**
On respecte ensuite une indentation, ie un décalage d'espaces par rapport à la ligne précédente : c'est très important sinon cela génère des erreurs.
On met alors des commentaires en écrivant entre """ Commentaires """
Il s'agit de décrire ce que fait la fonction, donner les types des variables d'entrées, ainsi que celui des variables de sortie.
On écrit ensuite les instructions opérant sur les variables d'entrées. On peut aussi commenter ces instructions en écrivant après #.
Elle donne en sortie zéro ou une variable de sortie. Dans ce cas, on utilise l'instruction **return**.
Les variables d'entrée et de sortie peuvent être des listes, ce qui permet en sortie de récupérer éventuellement plusieurs variables.

I.1 Syntaxe

La syntaxe ci-dessous crée une fonction :
•dont le nom est *ptsi*.
•qui utilise comme variables d'entrées $x_1, x_2, \ldots, x_n$.
•qui renvoie en sortie la variable y.

```
def ptsi(x_1, x_2, ..., x_n) :
    """ En commentaires, on explique ce que fait la fonction."""
    Instructions permettant de calculer y en utilisant x_1, x_2, ..., x_n
        # Commentaires sur les instructions
    return y
```

⚠ Il ne faut pas taper ce qui précède dans l'éditeur !

I.2 Exemples

Dans l'éditeur, ouvrez une nouvelle page, puis créez la fonction suivante :

```
def somme(a, b) :
    """ Cette fonction calcule et renvoie la somme de deux réels a et b."""
    s = a + b        # on stocke la valeur de a + b dans la variable s
    return s
    # On peut éviter l'utilisation de la variable s et se contenter de l'instruction return a + b
```

Enregistrez cette fonction dans un fichier, puis retournez dans la console, exécutez le fichier et tapez les instructions suivantes :

☞ *del a, b* ☞ *somme*(2, 3) ☞ *somme*(2) ☞ *somme*(2, 3, 5)
☞ $s = somme(2,3)$ ☞ *s* ☞ *a* ☞ *b*

Pensez à bien affecter une valeur à toutes les variables d'entrées et remarquez que celles-ci restent locales à la fonction. Notez qu'il est inutile de faire appel à l'instruction *input* dans le programme pour donner des valeurs aux variables d'entrées. Notez également que les variables *a* et *b* utilisées n'existent pas dans la console. Ce sont des variables **locales**.
Prenez également la bonne habitude de mettre le résultat de la fonction dans une variable.

Sautez une ligne dans la fenêtre de l'éditeur puis tapez ce qui suit après la première fonction :

```
x = 100
def mensonge() :
    """ Cette fonction ne prend aucun argument en entrée et renvoie 10."""
    x = 10        # variable locale à la fonction
    return x
```

Enregistrez cette fonction dans un fichier, puis retournez dans la console, exécutez le fichier et tapez les instructions suivantes :

☞ *x* ☞ *mensonge*() ☞ *x* **help** (*mensonge*)

La variable **globale** *x* (définie avant la fonction) n'a pas été modifiée par la fonction mensonge...
On accède à la documentation d'une fonction à l'aide de la commande **help** :

⚠ Une fonction peut contenir plusieurs fois la commande return, ce qui sera notamment le cas lorsque la fonction contiendra des tests. Toutefois, au premier return évalué, la fonction se termine.
Sautez une ligne dans la fenêtre de l'éditeur, tapez ## pour créer une nouvelle cellule puis créez ces nouvelles fonctions à la suite de la première :

```
1 def sommeproduit_bad(a, b) :
2     """ Cette fonction calcule et renvoie la somme et le produit de deux réels a et b."""
3     return a + b
4     return a * b
```

```
1 def sommeproduit(a, b = 4) :
2     """ Cette fonction calcule et renvoie la somme et le produit de deux réels a et b."""
3     t = a + b
4     return t, a * b
```

Enregistrez ces fonction dans votre fichier, puis retournez dans la console, exécutez le fichier (F5) ou la cellule (CTRL + Entrée) et tapez les instructions suivantes :

☞ *del s* ☞ *sommeproduit_bad*(2, 3) ☞ *sommeproduit*(2, 3)
☞ $s = sommeproduit(2)$ ☞ $s[0]$ ☞ $s[1]$

La première fonction ne renvoie que la somme des deux variables.
On peut utiliser le résultat d'une fonction dans une autre.
On peut donner une valeur par défaut à une variable d'entrée.
Si l'on a plusieurs résultats en sortie, ils sont stockés dans un **tuple**. On a accès aux valeurs du tuple en tapant les coordonnées entre crochet, la première coordonnée correspondant à l'indice 0

I.3 Utilisation de fonctions prédéfinies

De nombreuses fonctions sont définies dans PYTHON. Elles sont organisées par module. Le but de la formation est plutôt d'apprendre à créer soit même ses propres fonctions que de trouver celles de PYTHON susceptibles de résoudre notre problème. Néanmoins, certaines sont nécessaires (on ne va pas vous demander de fabriquer la fonction *cos*). Quand on connaît le nom d'une fonction, la commande $help(nom\_de\_la\_fonction)$ donne l'aide en ligne de cette fonction.
Si l'on veut faire des mathématiques, nous avons besoin du module "math" qui contient les principales fonctions mathématiques. En effet PYTHON ne connaît pas la fonction *cos* ou même le nombre π :
Tapez les instructions suivantes :

☞ $a = 4$ ☞ $cos(a)$ ☞ $b = exp(a)$ ☞ $c = pi$

•Le module "math"

Pour avoir accès aux fonctions contenues dans ce module, il faut l'importer. Trois méthodes sont possibles :
Tapez les instructions suivantes :

☞ *import math* ☞ $a = 4$ ☞ $math.sqrt(a)$ ☞ $math.cos(math.pi)$

Le "." signifie que l'on utilise une fonction ou une variable contenue dans le module math. On peut utiliser un alias en tapant : *import math as m*, et alors l'instruction $m.sqrt(a)$ est équivalente à $math.sqrt(a)$.
Pour avoir accès à toutes les fonctions de ce module, on tape dir(math), et pour obtenir une description de ces fonctions, il suffit de taper help(math).

Il existe donc une autre possibilité où l'on importe directement les fonctions que l'on veut utiliser :
Tapez les instructions suivantes :

☞ *from math import cos, pi, sqrt* ☞ $sqrt(a)$ ☞ $cos(pi)$

On peut aussi importer toutes les fonctions du module :

☞ *from math import ∗* ☞ $exp(0)$ ☞ $log(10)$ ☞ $log(e)$

Attention, log représente le logarithme népérien, la fonction ln n'a pas de signification sous PYTHON.

Comment caractériser ces 4 fonctions ?

•Le module random
Importer le module random dans la console : *import random as rd*.
Testez les fonction $random(), randint$ de ce module.

Notez que pour appliquer une fonction à un nombre ou une variable, il faut toujours mettre des parenthèses.

II Exercices

1. Ecrire une fonction qui étant donné 2 réels a et b, renvoie la différence $a - b$, le produit $a * b$ et le quotient a/b.
   Tester la fonction. Que se passe-t-il si l'on donne 0 comme valeur pour la deuxième variable ?
2. Moyenne de deux nombres
   (a) Écrire une fonction qui prend en arguments deux nombres et qui renvoie leur somme.
   (b) En utilisant la fonction précédente, écrire une fonction qui prend en entrées deux nombres et qui renvoie leur moyenne.
3. Écrire en Python la fonction suivante : $x \mapsto \dfrac{2x^5 + \ln(x)}{\sqrt{\pi} \times \cos(x)}$
   On pourra importer les fonctions dont on a besoin avec le module math.
4. Écrire une fonction qui prend en arguments une fonction f et un réel x, et qui renvoie la valeur de $f(x)$. On pourra la tester avec la fonction définie juste avant.
5. On considère un triangle rectangle dont les deux côtés perpendiculaires ont pour longueurs respectives a et b Ecrire une fonction qui étant donné ces 2 réels a et b, renvoie la longueur de l'hypothénuse.
   Tester cette fonction avec $a = 33$ et $b = 56$.
6. Écrire une fonction qui prend en entrées les coordonnées de deux points A et B du plan et qui renvoie la distance AB
7. Ecrire une fonction qui étant donné un entier n correspondant à une année de naissance renvoie l'âge de la personne née l'année n.
8. Ecrire une fonction qui étant donné une chaîne de caractère *nom* correspondant au prénom d'une personne renvoie une chaîne de caractère saluant cette personne.