Fiche 4 : Fonctions

1 Les fonctions

Les programmes vus jusqu'à maintenant présentent certains défauts : il nous est impossible d'utiliser un programme écrit précédemment pour l'utiliser dans un autre programme autrement qu'en faisant du "copier-coller" de code, ce qui le rend rapidement illisible si on devait écrire un long programme. De plus, si un autre utilisateur a besoin de notre programme il devra également effectuer un copier-coller. Ainsi, il nous manque la portabilité. Pour remédier à ce problème, on utilise les fonctions qui permettent d'une part de rendre un code plus lisible et d'autre part facilitent un travail en parallèle.

2 Fonctions à connaître

Voici une liste de quelques fonctions que nous avons déjà vu et qui sont à connaître (d'autres fonctions seront étudiées ultérieurement) :

Nom fonction	Effet	Exemple
`print`	affiche ce qui est en argument	print("toto")
`input`	laisse à l'utilisateur de saisir au clavier	a=input()
`str`	Convertit l'argument de la fonction en chaîne	v=str(1)
`int`	Convertit l'argument en entier si possible	v=int("12")
`float`	Convertit l'argument en réel si possible	v=float(input())

3 Syntaxe pour définir une fonction

Donnons quelques avantages des fonctions :
— possibilité d'automatisation de tâches ;
— possibilité de réutilisation ;
— valeur de retour stockable dans une variable.

fonction(arg1,$\cdots$,argN)
  Instructions
  retourner valeur
	`def` NomFonction(arg1,$\cdots$,argN) :
	  Instructions
	  `return` valeur

Parfois, on ne veut pas de valeur de retour et seulement sortir de la fonction. Dans ce cas, on peut simplement écrire `return`.

Exemples : minimum entre x et y et valeur absolue de x

```
def minimum(x,y) :
    if x<=y :
        return x
    else :
        return y
```

```
def absolue(x) :
    if x>=0 :
        return x
    else :
        return -x
```

4 Écrire un code lisible pour tous

Lorsque l'on écrit un programme informatique, il est courant qu'il soit seulement lisible par son auteur, et que même celui-ci, trois mois plus tard, ne comprend plus son propre code.

Pour remédier à ce problème il est IMPORTANT de commenter son code surtout s'il y a une tierce personne qui doit relire votre fichier. Voici quelques conseils pour une meilleure lisibilité du code :

1. donner à vos variables et à vos fonctions un nom qui a du sens ;
2. expliquer ce que fait votre fonction (description de l'algorithme s'il est complexe) ;
3. expliquer si besoin les points délicats à l'intérieur du programme (les conditions données, certaines instructions...).

Pour commenter, on écrit le texte correspondant entre""" """.

Exemple :

```
def minimum(x,y) :
""" calcule le minimum entre x et y"""
    if x<=y :        """ condition pour que x soit minimum """
       return x
    else :
       return y
```

Pour un commentaire assez court, on peut aussi utiliser le symbole # qui permet de commenter sur une ligne.

Exemple :

```
def sommePuissance(n,p) :
# Calculer la somme des n premiers puissance pieme
S=0
for i in range(1,n+1) :
    S=S+i**p
return S
```