

Chapitre 3

Fonction récursive

Version avec preuves.

Table des matières

1	Définition d'une fonction récursive	2
2	Utilisation de la pile	3

1 Définition d'une fonction récursive

Définition 1 :

Une fonction récursive est une fonction qui s'appelle dans la définition.

Pour assurer une terminaison à la fonction, il est nécessaire d'avoir dans la définition des cas terminaux qui ne nécessitent pas un autre appel.

Exemples 1 : Itératif vs récursif :

```
def compte_for (n) :  
    s = 0  
    for i in range(n) :  
        print(s)  
        s = s+1  
    return s  
  
print(compte_for(5))  
  
def compte_rec(n,i,s) :  
    if i == n :  
        return s  
    else :  
        print(s)  
        return compte_rec(n,i+1,s+1)  
  
print(compte_rec(5,0,0))
```

Exemples 2 : Suites définie par une relation de récursion en mathématiques :

La suite (u_n) définie pour tout $n \in \mathbb{N}$ par :
$$\begin{cases} u_0 = -1 \\ u_n = 2u_{n-1} - 3 \end{cases}$$

```
def u (n) :  
    if n == 0 :  
        return -1  
    else :  
        return 2*u(n-1) -3  
  
print(u(4))
```

Exemples 3 : Occurrence d'une lettre

```
def occurrence(a,mot) :  
    if mot == "" :  
        return 0  
    else :  
        if mot[0] == a :  
            return 1 + occurrence(a,mot[1:])  
        else :  
            return occurrence(a,mot[1:])  
  
print(occurrence("e","atredheaieojehzjfe"))
```

Exercice 1 :

1. Écrire la fonction `nb_de_chiffre` qui prend en argument un entier et compte le nombre de chiffres qui compose l'écriture de l'entier.
2. Écrire la fonction `nb_de_zero` qui prend en argument un entier et compte le nombre de 0 dans son écriture décimale.

Correction

```
def nb_de_chiffre(n) :  
    """ retourne le nb de chiffre """  
    if n == 0 :  
        return 0  
    else :  
        return 1 + nb_de_chiffre(n//10)  
  
nb_de_chiffre(152046)  
  
def nb_de_zero(n) :  
    """ retourne le nb de zero """  
    if n == 0 :  
        return 0  
    else :  
        if n%10 == 0 :  
            return 1 + nb_de_zero(n//10)  
        else :  
            return nb_de_zero(n//10)  
  
nb_de_zero(102550120)
```

2 Utilisation de la pile

Exemples 4 : Fonction factorielle

```
def fact(n) :  
    """ renvoie n! """  
    if n == 0 :  
        return 1  
    else :  
        return n * fact(n-1)
```

L'interpréteur gère une pile dite de récursion pour les appels successifs puis les calculs à effectuer.

Exercice 2 : Construire la pile pour le calcul de `fact(4)`.

Remarque : Une fonction récursive terminale est une fonction récursive où l'appel récursif est la dernière instruction à être évaluée. Il n'y a pas ici de phase de dépilement.

Exercice 3 : Écrire la fonction factorielle en version terminale.

Correction

```
def fact_terminale(n) :  
    def cumul(n,c) :  
        if n == 0 :  
            return c  
        else :  
            return cumul(n-1 , n*c)  
    return cumul(n,1)  
  
print(fact_terminale(5))
```
