

2.2

Savoir construire un programme récursif

NSI TERMINALE - JB DUTHOIT

2.2.1 Exemple 1 : puissance d'un entier

Exercice 2.71

CONSTRUIRE UNE FONCTION RÉCURSIVE REVOYANT LA PUISSANCE D'UN ENTIER NATUREL

Soit a un entier non nul et n un entier naturel.

En sachant que $a^0 = 1$ et $a^n = a^{n-1} \times a$, construire la fonction récursive `fonction(a,n)` qui prend comme paramètre `a` et `n` et qui retourne le résultat de a^n .

2.2.2 Exemple 2 : somme des entiers consécutifs

Exercice 2.72

SAVOIR CALCULER LA SOMME DES PREMIERS ENTIERS CONSÉCUTIFS.

Construire une fonction récursive `somme(n)` qui permet de calculer :

$$0 + 1 + 2 + 3 + \dots + n$$

2.2.3 Longueur d'une liste

Exercice 2.73

Créer une fonction récursive `long(t)` avec comme paramètre `t` un tableau dynamique python et qui renvoie la longueur du tableau `t`. Pour rappel au niveau du slicing :

```
t = [1,2,3,4,5,6,7]
t[1:4] # est le tableau avec les éléments de l'indice 1
       inclus à l'indice 4 exclus ([2,3,4])
t[:4] # est le tableau avec les éléments jusqu'à l'indice 4
       exclus ([1,2,3,4])
t[3:] # est le tableau avec les éléments à partir de l'
       indice 3 inclus ([4,5,6,7])
```

2.2.4 Exemple 3 : nombre d'occurrence(s)

Exercice 2.74

SAVOIR CALCULER LE NOMBRE D'OCCURRENCES D'UN CARACTÈRE DANS UNE CHAÎNE.

Construire une fonction récursive `occurrenec(chaine,caractere)` qui prend en paramètre deux chaînes de caractères `chaine` et `caractere`. La fonction renvoie le nombre d'occurrence(s) de `caractere` dans `chaine`.

- Le nombre d'occurrences de `caractere` dans `chaine` est 0 si `chaine` est vide.
- Si `caractere` est le premier caractère de `chaine`, on ajoute 1 au nombre d'occurrences de `caractere` dans les autres caractères de `chaine`
- Sinon, il s'agit du nombre d'occurrences de `caractere` dans les autres caractères de `chaine`.

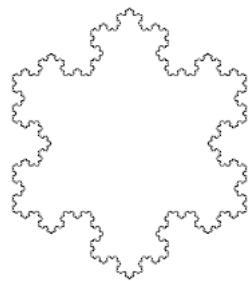
Avant de réaliser cet exercice, on créera une fonction `sansdebut(chaine)` avec comme paramètre `chaine` une chaîne de caractère, et qui renvoie la même chaîne de caractère sans le premier caractère. Ainsi `sansdebut("voiture")` donnera "oiture". Pour créer cette fonction `sansdebut(chaine)`, on s'interdira d'utiliser le slicing.

2.2.5 Exemple 4 : le flocon de Von Koch

Exercice 2.75

SAVOIR CONSTRUIRE UN PROGRAMME RÉCURSIF POUR DESSINER (PLUS DIFFICILE)

Il s'agit ici de construire un programme qui permet de dessiner le flocon de Von Koch :



Il faut commencer à vous familiariser avec le module `turtle` de Python, qui permet de dessiner facilement.

Tester et analyser ce petit programme :

```
from turtle import *
forward(100)
right(90)
forward(100)
right(90)
forward(100)
right(90)
forward(100)
right(30)
forward(100)
right(120)
forward(100)
```

Puis il faut approfondir le sujet avec : Wikibook sur le module `turtle` Python

Et enfin se familiariser avec le flocon de Von Koch en visionnant cette vidéo : Vidéo sur le flocon

2.2.6 Conclusion

La récursivité offre donc au programmeur un autre moyen, souvent élégant et concis, de résoudre des problèmes.

Par exemple :

- dans la programmation des jeux solitaires du type Sudoku, labyrinthes...
- dans la programmation des jeux à deux joueurs du type Échecs, Dames, Othello...
- et dans bien d'autres domaines encore