2.3 Savoir construire un programme récursif

NSI Tle - JB Duthoit

2.3.1 Exemple 1 : puissance d'un entier

Exercice 2.26
Construire le programme récursif de 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 le programme récursif qui prend comme paramètre a et n et qui retourne le résultat de a^n.

2.3.2 Exemple 2 : somme des entiers consécutifs

Exercice 2.27
Savoir calculer la somme des premiers entiers consécutifs.
Construire un programme récursif qui permet de calculer :

$$0 + 1 + 2 + 3 + ... + n$$

2.3.3 Longueur d'une liste

Exercice 2.28
***Imaginer un programme récursif qui retourne la longueur d'une liste.
On pourra proposer plusieurs versions :

- Une version avec les slides (Hors programme de NSI)
- Une version avec une la fonction `extraction`.
- Une version avec la compréhension de liste
- une version avec un nouveau paramètre

2.3.4 Exemple 3 : nombre d'occurrence(s)

Exercice 2.29
Savoir calculer le nombre d'occurrences d'un caractère dans une chaîne.

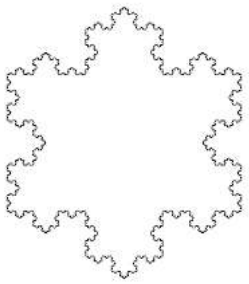
- Le nombre d'occurrences de c dans s est 0 si s est vide.
- Si c est le premier caractère de s, on ajoute 1 au nombre d'occurrences de c dans les autres caractères de s
- Sinon, il s'agit du nombre d'occurrences de c dans les autres caractères de s.

2.3.5 Exemple 4 : le flocon de Von Koch

Exercice 2.30

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

2.3.6 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