TD Maple n°5 : Récursivité
Lycée Louis le Grand
PCSI 1

Lundis 2 et 9 février 2009

1 Définitions et exemples

1.1 Rappel sur les procédures

On rappelle qu'une procédure est un objet Maple qui prend en *argument* une liste de paramètres, et qui *renvoie* un résultat. Nous avons déjà eu l'occasion de constuire, par exemple, des procédures qui prenaient comme arguments un entier n, et un réel u_0, et qui renvoyaient le n-ième terme d'une suite récurrente $u_{n+1} = f(u_n)$. On rapelle que la syntaxe générale pour définir une procédure est la suivante :

```
[>nom_procedure :=proc(parametre_1,parametre_2,...,parametre_n)
local variable_locale_1,...,variable_locale_p ;
global variable_globale_1,...,variable_globale_q ;
intruction_1 ;
.
.
instruction_m ;
end proc ;
```

Une variable globale est, par opposition à une variable locale, connue dans toute la session Maple en cours, et pas seulement à l'intérieur de la procédure. Le résultat renvoyé par la procédure est la dernière instruction exécutée à l'intérieure de celle-ci. Dans l'exemple ci-dessus, il s'agit de l'instruction `instruction_m ;`.

1.2 Procédures récursives

Une procédure est dite *récursive* lorsqu'elle s'appelle elle-même. Les procédures récursives sont les équivalents, en programmation, des définitions par récurrence. Une fonction est définie par récurrence lorsqu'on se donne :

- sa valeur lorsque son argument est minimal,
- une formule de récurrence qui permet de calculer la valeur de la fonction pour un argument à partir de ses valeurs sur des arguments strictement plus petits.

De même, une procédure récursive comportera :

- tout d'abord un test pour déterminer si l'argument est minimal et donner dans ce cas le résultat, c'est ce qu'on appelle le *cas d'arrêt*,
- ensuite, le traitement du cas général qui contient l'appel récursif. Il est impératif que l'appel récursif soit fait sur un argument strictement plus petit, sous peine de "boucler" indéfiniment.

L'un des avantages des programmes récursifs, est qu'ils sont souvent faciles à obtenir par simple traduction de la définition mathématique récursive. Par exemple, la factorielle est définie par $0! = 1$ (cas d'arrêt), et pour $n > 0$, $n! = n \times (n-1)!$. La traduction en Maple est immédiate :

```
[>factorielle :=proc(n)
if n=0 then 1 else n*factorielle(n-1) ; fi ;
end proc ;
```

2 La suite de Fibonacci

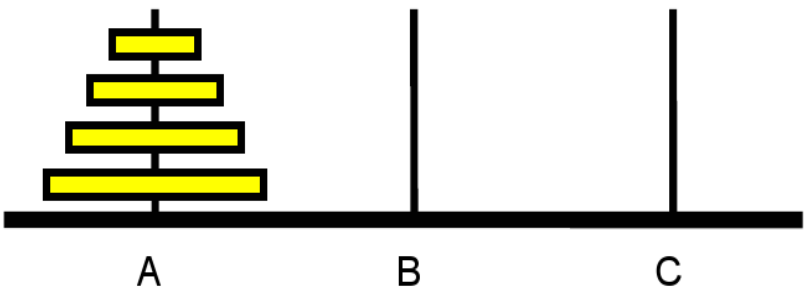
Les fonctions récursives sont parfois inefficaces pour cause de calculs inutilement répétés. Nous allons voir un exemple de ce problème sur le calcul des termes de la suite de Fibonnacci définie par :

$$\begin{cases} u_0 = 1, u_1 = 1 \\ \forall n \in \mathbb{N}, \quad u_{n+2} = u_{n+1} + u_n \end{cases}$$

1. Ecrire une procédure `fibo(n)` qui calcule u_n de manière récursive.
2. Pour un entier $n \in \mathbb{N}$ donné, combien d'appels à la fonction `fibo` sont effectués afin de calculer u_n ? (*On pourra raisonner par récurrence.*)
3. Dans le TD n°3 sur les suites, nous avons calculé les nombres u_n de manière *itérative*. Pour $n \in \mathbb{N}$ donné, quel est le nombre d'opérations élémentaires effectuées par cette méthode pour calculer u_n ? (*On appelle opération élémentaire une addition, une multiplication, une comparaison ou une affectation.*)
4. Conclure qu'une procédure récursive est peu efficace pour calculer les nombres u_n car elle mène à faire plus d'opérations que nécessaire.
5. Avec Maple, on peut palier à ce problème en ajoutant l'`option remember` à la procédure. Ceci oblige Maple à garder en mémoire chaque résultat de la procédure, pour que lors d'un futur appel identique, le calcul ne soit pas effectué à nouveau. Ajouter l'`option remember` à votre procédure et comparer avec ce qui précède.

3 Les tours de Hanoï ou quand la fin du monde adviendra-t-elle ?

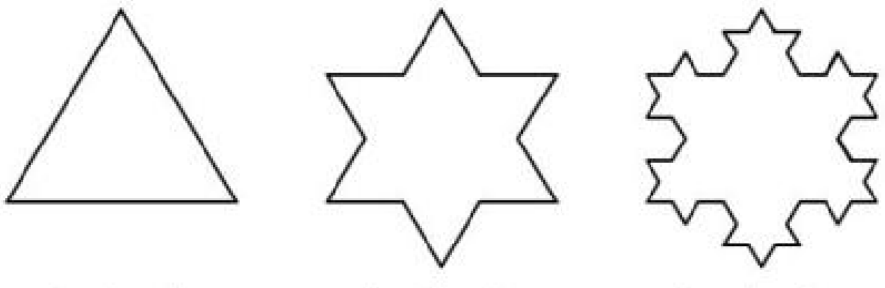
Le mathématicien français Edouard Lucas (1842-1891), professeur aux lycées Saint-Louis et Charlemagne, raconte qu'un des ses amis a vu, lors d'un voyage, un temple hindou où des prêtres sont astreints à la tâche suivante : déplacer des disques de la tour A vers la tour C en utilisant si besoin est la tour B. Ils ne déplacent qu'un seul disque à la fois, et de telle façon qu'un disque ne vienne jamais se placer sur un autre de diamètre inférieur.



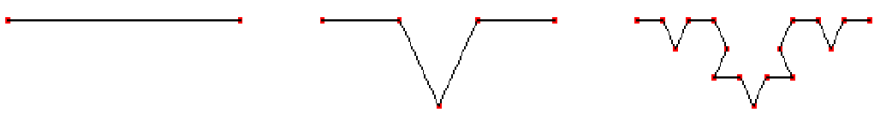
1. Ecrire une procédure récursive qui imprime la liste des déplacements (phrase du type "déplacer disque de B vers C ") pour un nombre n quelconque de disques. La procédure `Hanoi(n,A,B,C)` reçoit le nombre n de disques à déplacer de la tour A vers la tour C en utilisant la tour intermédiaire B.
   On ramarquera que pour déplacer les n disques de la tour A vers la tour C, on s'y prend de la manière suivante :
   - Déplacer les $n-1$ disques supérieurs de la tour A vers la tour B,
   - Déplacer le dernier disque (le plus gros) de la tour A vers la tour C,
   - Déplacer les $n-1$ disques qui sont sur la tour B vers la tour C.
2. Exécuter la procédure pour $n = 4$, $n = 5$. (Ne pas aller trop loin !)
3. Le nombre de disques étant de 64, et les prêtres en déplaçant un par seconde (des sprinters du disque), on raconte que leur tâche durera jusqu'à la fin du monde. En supposant qu'ils aient commencé il y a 200 000 ans (apparition d'*homo sapiens*), combien de temps reste-il avant l'apocalypse ?
   On pourra calculer, toujours par récurrence, le nombre de déplacements nécessaires pour déplacer n disques de la tour A vers la tour C.
4. Modifier la procédure `Hanoi` pour qu'elle calcule le nombre de déplacements effectués lorsqu'on exécute la procédure. (*Attention, la variable contenant ce nombre devra être déclarée comme variable globale !*) Vérifier que le résultat est cohérent avec la question 3.

4 Le flocon de Von Koch

Le flochon de Von Koch est une courbe fractale obtenue par le procédé suivant : A l'étape 0, on part d'un triangle équilatéral[1], puis on divise chaque côté en trois segments de même longueur et on remplace chaque segment central par deux cotés d'un triangle équilatéral construit extérieurement à partir de ce segment central. On itère ensuite le processus.



1. On commence par construire la fractale correspondant à un seul coté du triangle de départ :



   Pour dessiner le segment de départ avec Maple, on construit une séquence N constituée des coordonnées des deux sommets du segment, puis on utilise l'instruction `plot` :

```
[>N :=[0,0],[1,0] ;
[>plot([N],axes=none) ;
```

[1] On rappelle q'un triangle équilatéral est un triangle dont les trois côtés sont de même longueur.

Créer alors une procédure récursive `vonkoch(n,xA,yA,xB,yB)` qui, à partir d'un segment initial dont les extrémités A et B ont pour coordonnées respectives $[xA, yA]$ et $[xB, yB]$, construit la séquence des coordonnées des points constituant la figure à l'étape n.

2. Testez votre procédure (Ne pas dépasser $n = 7$).
3. Pour dessiner le triangle de départ avec Maple, on construit une séquence N constituée des quatres sommets du triangle (si si j'ai bien dit quatre) à relier par des segments, puis on utilise l'instruction `plot` comme suit :

```
[>N :=[0,0],[1,0],[1/2,sqrt(3)/2],[0,0] ;
[>plot([N],axes=none) ;
```

Construire alors le flocon de Von Koch à l'étape n.

5 Le triangle de Sierpinski

Maintenant que vous êtes des pros de la fractale, vous pouvez dessiner le triangle de Sierpinski.