Programmation en C et structures de données

guillaume.revy@univ-perp.fr

Fonctions récursives

Exercice 1. Réinventons la roue

On dispose d'une machine ne sachant faire que des additions/soustractions par 1 et on souhaite implanter l'addition, la soustraction et la multiplication de deux nombres entiers, notés ici a et b.

► **1.** Écrire une fonction récursive addition qui calcule $a+b$ en utilisant que des additions/soustractions par 1.

► **2.** Sur le même modèle, écrire une fonction récursive soustraction qui calcule $a - b$ en utilisant que des additions/soustractions par 1.

► **3.** En utilisant les fonctions précédentes, écrire une fonction récursive multiplication qui calcule $a \times b$.

Exercice 2. Nombre de Ackermann

► **1.** Écrire une fonction récursive qui calcule le nombre d'Ackermann, défini sur $\mathbb{N}\times\mathbb{N}$ de la manière suivante :

$$A(m,n) = \begin{cases} n+1 & \text{si } m = 0, \\ A(m-1,1) & \text{si } m > 0 \text{ et } n = 0, \\ A\big(m-1, A(m,n-1)\big) & \text{si } m > 0 \text{ et } n > 0. \end{cases}$$

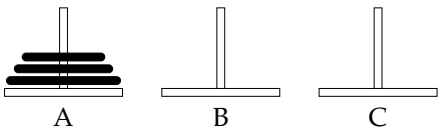
Exercice 3. Tours de Hanoï

Le problème des *Tours de Hanoï* est un problème mathématique (Édouard Lucas, 1883) qui consiste à trouver une stratégie pour déplacer une pile de n disques de diamètres tous différents, d'un socle A vers un socle C en passant par un socle intermédiaire B, en un minimum de coups, et tout en respectant les règles suivantes :

— on ne peut déplacer qu'un disque à la fois,

— et on ne peut placer un disque que sur un autre disque plus grand que lui ou sur un emplacement vide.

Au départ, les disques sont tous empilés sur le socle A par diamètre décroissant, comme indiqué sur la figure ci-dessous, pour $n = 3$.



► **1.** Définir une structure `socle_t` qui permet de représenter un socle, et qui contient notamment : l'indice (A, B ou C) du socle dans le problème, un tableau contenant le diamètre des disques présents sur le socle, le nombre courant de disques sur le socle et le nombre maximum de disques que l'on peut empiler.

► **2.** Écrire une fonction de création/initialisation des socles, une de destruction, et une d'affichage.

► **3.** Écrire un programme principal, qui lit la valeur de n, créé et initialise les socles A, B et C, puis les détruit.

Le problème des *Tours de Hanoï* peut être résolu récursivement grâce à l'algorithme `Hanoi` suivant :

`Hanoi(n, src, inter, dest)` =

$$\begin{cases} \text{si } n = 1 & \text{déplacer ce disque du socle } \texttt{src} \text{ vers le socle } \texttt{dest}, \\ \text{si } n \neq 1 & \text{(1) déplacer } (n-1) \text{ disques du socle } \texttt{src} \text{ vers le socle } \texttt{inter} \text{ en passant par le socle } \texttt{dest}, \\ & \text{(2) déplacer 1 disque du socle } \texttt{src} \text{ vers le socle } \texttt{dest}, \\ & \text{(3) et déplacer } (n-1) \text{ disques du socle } \texttt{inter} \text{ vers le socle } \texttt{dest} \text{ en passant par le socle } \texttt{src}. \end{cases}$$

Le premier appel sera : `Hanoi(n,A,B,C)`.

► **4.** Écrire finalement la fonction récursive Hanoi, et afficher les différentes étapes de son exécution sous forme graphique.