Lycée Saint-Louis TP
MPSI

Syntaxe et récursivité en Caml

Pour chaque fonction récursive, on vérifiera la terminaison et la correction.

1. Écrire une fonction `expo_rapide_terminale` prenant en argument un réel x et un entier n et renvoyant x^n en utilisant l'algorithme d'exponentiation rapide qui sera codé de façon récursive terminale.

2. On considère une suite u telle que $\forall n \in \mathbb{N},\ u_{n+1} = \begin{cases} u_n/2 & \text{si } u_n \text{ est pair} \\ 3u_n + 1 & \text{si } u_n \text{ est impair} \end{cases}$

   La *conjecture de Syracuse* est une proposition non démontrée à ce jour qui affirme que la suite u atteint toujours la valeur 1 pour tout $u_0 \in \mathbb{N}^*$.

   (a) Écrire une fonction `syracuse` qui prend en paramètre deux entiers n et p et qui renvoie u_n lorsque $u_0 = p$.

   (b) Écrire une fonction `affiche` qui prend en paramètre deux entiers n et p et qui affiche les $n+1$ premiers termes de la suite lorsque $u_0 = p$.
   On utilisera la commande `print_newline ()` pour passer à la ligne.

   (c) Écrire une fonction `tempsdeVol` qui prend en paramètre un entier p et qui renvoie le plus petit entier naturel n tel que $u_n = 1$ lorsque $u_0 = p$. Tester plusieurs données initiales. Si ce n'est pas le cas, la rendre terminale.

   (d) Écrire une fonction `hauteur` qui prend en paramètre un entier p et qui renvoie la plus grande valeur prise par la suite u. Si ce n'est pas le cas, la rendre terminale.

   (e) Écrire une fonction `somme` qui prend en paramètre deux entiers n et p et qui renvoie $\sum_{k=0}^{n} u_k$ lorsque $u_0 = p$. Si ce n'est pas le cas, la rendre terminale.

3. Les tours de Hanoï sont un jeu constitué de trois tiges sur lesquelles on peut enfiler des disques.
   Initialement, on a placé n disques de diamètres différents placés du plus grand au plus petit sur la première tige. On souhaite arriver à la même construction mais sur la dernière tige en respectant deux règles :
   — on ne déplace qu'un disque à la fois
   — on ne peut déplacer un disque sur un disque plus grand.

   (a) Écrire une fonction `hanoi` prenant en paramètre un entier n et affichant la suite des opérations à réaliser. Pour $n = 2$, on affichera :
   déplacement de la tige 1 vers la tige 2
   déplacement de la tige 1 vers la tige 3
   déplacement de la tige 2 vers la tige 3

   (b) Déterminer le nombre de déplacements effectués pour résoudre le problème avec n disques ?

   (c) On rajoute une nouvelle règle : tout mouvement doit concerner la tige du milieu, c'est-à-dire que soit on enlève un disque de la tige du milieu soit on déplace un disque sur la tige du milieu.

      i. Écrire une fonction `hanoi2` prenant en paramètre un entier n et affichant la suite des opérations à réaliser.

      ii. Déterminer le nombre de déplacements effectués pour résoudre le problème avec n disques.

4. (a) Écrire une fonction `partitions` prenant en argument deux entiers non nuls n et p et renvoyant le nombre de partitions de n en p entiers c'est-à-dire le cardinal de

$$\mathcal{P}_{n,p} = \{(m_1, ..., m_p) \in (\mathbb{N}^*)^p \ :\ \sum_{k=1}^{p} m_k = n \text{ et } m_1 \leq m_2 .. \leq m_p\}.$$

   On trouvera une relation de récurrence en faisant une disjonction de cas suivant la valeur de m_1.

   (b) Écrire une fonction `partitions2` prenant en argument un entier n et renvoyant le nombre de partions de n. Si ce n'est pas le cas, la rendre terminale.

5. On considère le jeu suivant : on dispose de n jetons positionnés sur des cases numérotés de 1 à n. L'objectif est de les retirer en respectant trois règles :
   — on ne peut mettre qu'un pion par case ;
   — on peut enlever ou mettre un pion dans la première case ;
   — on ne peut modifier le contenu de la case $c \in [\![2, n]\!]$ que lorsque la case $c-1$ est occupée et les cases précédentes sont libres.

   (a) Écrire une fonction `solution` prenant en paramètre un entier n et affichant la suite des opérations à réaliser. Pour $n = 3$, on affichera :
   retirer le pion de la case 1
   retirer le pion de la case 3
   mettre un pion dans la case 1
   retirer le pion de la case 2
   retirer le pion de la case 1

   (b) Déterminer le nombre de mouvements effectués pour résoudre le problème avec n cases.