DS 0

Option informatique, deuxième année

Julien Reichert

Durée : environ une heure et demie

Exercice 1 : Écrire en Caml un algorithme de tri au choix. Celui-ci peut être en place ou non, et agir sur des listes ou des tableaux.

Exercice 2 : Écrire en Caml une fonction qui prend en argument deux valeurs `borneinf` et `bornesup` et qui retourne le couple `(n, m)` tel que la somme des diviseurs de `n` et celle des diviseurs de `m` soient toutes les deux égales à `m+n+1`, en imposant que `m` soit strictement supérieur à `n` et que `n` soit le plus petit entier supérieur ou égal à `borneinf` vérifiant la propriété ci-avant. Si le nombre `n` en question est strictement supérieur à `bornesup`, la fonction doit déclencher une erreur.

Exercice 2bis : Déterminer la complexité temporelle de la fonction précédente dans le pire des cas avec le plus de rigueur possible (ce n'est pas forcément évident).

Exercice 3 : Soit un tableau d'entiers dont on note `n` la taille et on garantit que les valeurs sont des entiers compris entre `1` et `n-1`, donc qu'au moins une valeur se répète. Écrire en Caml une fonction qui prend en entrée un tableau sous ces conditions (qu'on ne vérifiera pas) et qui détermine une valeur qui se répète. Il n'y a aucune contrainte sur la complexité.

Exercice 3bis : Déterminer la complexité temporelle de la fonction précédente dans le pire des cas.

Exercice 4 : Soit un tableau d'entiers contenant tous les nombres de `1` à une certaine valeur `n` et uniquement de tels nombres. L'un des nombres figure au moins deux fois, chacun des autres figure exactement une fois. Écrire en Caml une fonction qui prend en entrée un tableau sous ces conditions (qu'on ne vérifiera pas) et la valeur `n` et qui détermine la valeur qui se répète. L'objectif est que le programme ait une complexité en espace constante et une complexité en temps linéaire. Il sera également interdit de modifier le tableau.

Exercice 5 (très difficile) : Soit un tableau d'entiers de taille `n+1`, dont les éléments sont tous des nombres entre `1` et `n`, mais pas forcément tous présents. Un seul des nombres figure au moins deux fois. Écrire en Caml une fonction qui prend en entrée un tableau sous ces conditions (qu'on ne vérifiera pas) et qui détermine la valeur qui se répète. Le programme ne sera accepté que si sa complexité en espace est constante, et sa complexité en temps est linéaire. Il sera également interdit de modifier le tableau.

Hé, c'est un secret, mais il y a un verso avec des indications pour l'exercice 5 !

Un algorithme relativement classique se révèle particulièrement efficace pour l'exercice 5, il s'agit de l'algorithme du lièvre et de la tortue, introduit par Robert Floyd.

Procédons par étapes.

1) On considère la suite $(t_n)_{n\in\mathbb{N}}$ telle que $t_0 = 0$ et pour tout entier naturel i, t_{i+1} est la valeur de la case d'indice t_i du tableau. Pourquoi cette suite est-elle bien définie ? Pourquoi y a-t-il une répétition dans cette suite ? Pourquoi, dans les conditions restreintes sur le tableau, cette répétition est-elle unique et existe-t-il $i < j$ tels que $t_i \neq t_j$ mais $t_{i+1} = t_{j+1}$? (On pourra se demander pourquoi cette dernière propriété, qui est essentielle pour la résolution de l'exercice, ne serait pas forcément vraie si des éléments du tableau pouvaient être nuls, même en maintenant la garantie qu'un élément exactement se répète, en exhibant un contre-exemple.)

2) Une fois qu'on a établi l'existence des i et j ayant la propriété évoquée à la fin du point précédent, cela permet de déduire qu'il existe un cycle de taille $j - i$ qui démarre en i ou avant (en l'occurrence au plus tôt en $i - (j - i)$ si cette valeur est positive). Pour le détecter, on peut introduire une suite $(l_n)_{n\in\mathbb{N}}$ telle que $l_n = t_{2n}$. Dans ce cas, puisqu'il existe un i tel que $t_i = l_i$, dès qu'on aura trouvé une telle valeur i, on aura trouvé un cycle de taille i.

3) Soit maintenant la suite $(s_n)_{n\in\mathbb{N}}$ telle que $s_n = t_{i+n}$, pour la valeur i calculée grâce au calcul simultané des premières valeurs des deux suites. Cette nouvelle suite a i valeurs d'avance sur la suite $(t_n)_{n\in\mathbb{N}}$. Une fois que l'autre entre dans le cycle, c'est forcément sur la valeur qui se répète, car elle a deux prédécesseurs différents, et cela se verra car ce sera le premier indice k tel que $s_k = t_k$.

Il n'y a plus qu'à formaliser ces calculs sous la forme d'un algorithme puis d'un programme en Caml.