Option Informatique

Devoir Surveillé *n*°02

Durée : 1h. Le 22/04/2024.

Tout matériel électronique et tout document est interdit. Longueur : 1 page.

Exercice 1 : Manipulation de matrices

Dans ce problème on implémente les matrices d'entiers par des tableaux de tableaux, par exemple la matrice 2×3 $\begin{pmatrix} a & b & c \\ d & e & f \end{pmatrix}$ par le tableau `[|[|a;b;c|];[|d;e;f|]|]`.

1. Écrire une fonction `produit : int array array → int array array → int array array` telle que `produit a b` renvoie la matrice `a`×`b`. Résultat arbitraire si les dimensions des matrices ne sont pas compatibles.

2. En supposant que a est de dimension $n \times p$ et b de dimension $p \times q$, combien d'additions et combien de multiplications un tel appel effectue-t-il ? Donner une **brève** justification.

3. Écrire une fonction récursive `puissance : int array array → int → int array array` telle que `puissance m n` renvoie m^n, calculé à l'aide de l'algorithme d'exponentiation rapide.

4. Justifier que, pour des matrices $p \times p$, cette fonction effectue $O(p^3 \lg(n))$ calculs d'additions et de multiplications d'entiers.

Exercice 2 : Arbres binaires (non nécessairement stricts)

On définit le type `arbre_binaire` comme suit :

```
type 'a arbre_binaire = Vide | Noeud of 'a * 'a arbre_binaire * 'a arbre_binaire;;
```

5. Écrire en Caml une fonction `etiquette : 'a arbre_binaire → 'a`, puis des fonctions `fils_gauche` et `fils_droit : 'a arbre_binaire → 'a arbre`. Elle devront provoquer une exception en dehors de leur domaine de définition.

6. Écrire en Caml une fonction `taille : 'a arbre_binaire → int` permettant de calculer la taille de son argument.

7. Écrire en Caml une fonction `hauteur : 'a arbre_binaire → int` permettant de calculer la hauteur de son argument.

8. On rappelle qu'une feuille est un nœud dont les deux fils sont vides. Écrire en Caml une fonction `liste_feuille : 'a arbre_binaire → 'a list` permettant d'obtenir la liste des étiquettes des **feuilles** de son argument. La complexité de cette fonction pourra être sous-optimale et en particulier on n'hésitera pas à faire usage de la concaténation @.