Arbres

■ ■ ■

Les arbres généraux `'a arbre` et les arbres binaires `bin` étiquetés sont représentés par les types suivants. Une feuille est un noeud dont la liste des enfants est vide, ou un noeud dont les deux sous-arbres sont `Vide`.

```
type 'a arbre = Noeud of 'a * ('a arbre list)

type 'a bin = Vide | Nd of 'a * ('a bin) * ('a bin);;
```

Exercice 1. (Arbres généraux)

1. Écrire une fonction `nb_feuilles` qui renvoie le nombre de feuilles d'un arbre.
`nb_feuilles : 'a arbre -> int`.

2. Écrire une fonction `hauteur` qui renvoie la hauteur d'un arbre.
`hauteur : 'a arbre -> int`.

3. Montrer que l'ensemble des arbres généraux non étiquetés et ordonnés possédant n nœuds est en bijection avec l'ensemble des arbres binaires à $n-1$ nœuds. Illustrer cet algorithme sur des exemples.
Indication : Mettre les enfants à gauche, les frères à droite.

Exercice 2. (Cheminement) Le *cheminement* d'un arbre est la somme des profondeurs de chacune de ses feuilles. Écrire une fonction `cheminement` qui calcule le cheminement d'un arbre binaire non étiqueté.
`cheminement : bin -> int`

Exercice 3. (Ordre préfixe) Écrire une fonction `liste_prefixe` qui constitue la liste des étiquettes des nœuds d'un arbre étiqueté suivant l'ordre préfixe.
`'a arbre -> 'a list`

Exercice 4. (Sous-arbre) Soient T et T' deux arbres binaires étiquetés.

1. Écrire une fonction `est_sous_arbre` qui détermine si l'arbre binaire T est un sous-arbre de l'arbre binaire T'.
`est_sous_arbre : bin -> bin -> bool`

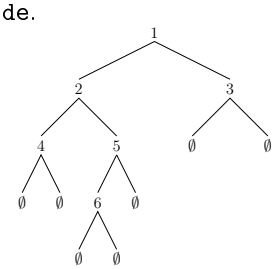
2. Écrire une fonction `est_inclus` qui détermine si l'arbre binaire T est inclus dans l'arbre binaire T', i.e. peut-être obtenu à partir de T' en supprimant certaines branches.
`est_inclus : bin -> bin -> bool`

Exercice 5. (Arbres parfaits)
Un arbre binaire est *parfait* si, à toute profondeur, tous ses nœuds sont d'arité 2, sauf éventuellement le dernier niveau. Écrire une fonction booléenne `est_parfait` qui détermine si un arbre est parfait. Évaluer la complexité de cette fonction.
`est_parfait : bin -> bool`

Exercice 6. On, considère la fonction `f` définie inductivement sur les arbres binaires non étiquetés par

```
let rec f a1 a2 =
  match a1 with
  | Vide -> a2
  | Nd (a3, a4) -> f a3 (Nd (a2, a4))
;;
```

1. On considère l'arbre `A` suivant (dont les nœuds ont été numérotés pour une meilleure compréhension). Calculer `f A Vide`, noté `B`, puis `f B Vide`.



2. Montrer que la fonction termine quels que soient les arbres passés en entrée.

3. Montrer, pour tout arbre `A`, l'égalité `f (f A Vide) Vide = A`.
On pourra montrer que `f (f A B) Vide = f B A`.

Exercice 7. (Dénombrement) On note B_n le nombre d'arbres binaires non étiquetés à n nœuds.

1. Décrire l'ensemble des arbres binaires à 0, 1, 2, 3 puis 4 nœuds et en déduire les valeurs de $B_0, \dots, B_4$.

2. Montrer que (B_n) est solution de l'équation de récurrence

$$B_0 = 1,\ B_{n+1} = \sum_{k=0}^{n} B_k B_{n-k}.$$

On rappelle que l'ensemble $\mathscr{B}_n$ des arbres binaires à n nœuds est en bijection avec l'ensemble $\mathscr{G}_{n+1}$ des arbres généraux à $n+1$ nœuds. On propose de calculer B_{n+1} en utilisant les chemins de Dyck.

3. On considère la fonction suivante qui à tout arbre non étiqueté associe la suite suivante.

```
let rec contour arbre =
  let rec aux foret h =
    match foret with
    | [] -> []
    | Noeud(a,q1)::q ->
      let c1 = aux q1 (h+1) and c2 = aux q h in
      [h] @ c1 @ [h-1] @ c2
  in aux [arbre] 0;;
```

a) Représenter graphiquement la fonction contour.

b) Étant donné un arbre T à n nœuds, calculer la longueur de la liste retournée par `contour` T.

c) Soit t un arbre à n nœuds et $[a_0; \dots; a_{2n-1}] =$ `contour` t le mot associé. Montrer que pour tout entier naturel $k \in [\![1, 2n-2]\!]$, $a_k \geqslant 0$ avec égalité si $k = 2n-2$.

Un chemin de Dyck de longueur n est une suite $(\gamma_0, \dots, \gamma_n)$ d'entiers positifs telle que $\gamma_0 = \gamma_n = 0$ et pour tout $k \in [\![0, n-1]\!]$, $|\gamma_{k+1} - \gamma_k| = 1$.

4. Montrer que $B_n = D_{2n}$.

5. Dans cette question, on souhaite montrer que $D_{2n} = \frac{1}{n+1}\binom{2n}{n}$. Soit $\Gamma_{2n}(a,b)$ l'ensemble des chemins de longueur $2n$, de pas appartenant à $\{-1,1\}$ tels que $\gamma_0 = a$ et $\gamma_{2n} = b$.

a) Déterminer $|G_{2n}(a,b)|$.

b) On note $\Gamma_{2n}^{\geqslant 0}$ (resp. $\Gamma_{2n}^{>0}$) l'ensemble des chemins tels que pour tout $i \in [\![0, 2n]\!]$, $\gamma_i \geqslant 0$ (resp. > 0) et Γ_{2n}^{0} l'ensemble des chemins qui rencontrent l'axe des abscisses. Montrer (on pourra utiliser des dessins) que

$$\begin{aligned}\left|\Gamma_{2n}^{\geqslant 0}(0,0)\right| &= \left|\Gamma_{2n}^{>0}(1,1)\right| \\ &= |\Gamma_{2n}(1,1)| - \left|\Gamma_{2n-2}^{0}(1,1)\right| \\ &= |\Gamma_{2n}(1,1)| - |\Gamma_{2n}(-1,1)| \\ &= \frac{1}{n+1}\binom{2n}{n}.\end{aligned}$$

L'entier $C_n = \frac{1}{n+1}\binom{2n}{n}$ *est le* n*-ème nombre de* Catalan.

6. Conclure.