Arbres

Dans tout le sujet, on considèrera des arbres. La structure de donnée est la suivante : de la même façon que pour les listes chainées, on utilise une structure basée sur des cellules, appelées ici *noeuds*, et chaque cellule a deux attributs :

— une valeur *val* (ici ce sera sauf mention du contraire un entier)

— une liste ordonnée de noeuds appelés *fils*.

On appelle *feuille* un noeud dont la liste de fils est vide. Un noeud *interne* est un noeud qui n'est pas une feuille.

Les arbres dits *binaires* sont ceux où chaque noeud a au plus deux fils. Dans ce cas, on n'utilisera pas une liste pour les fils mais deux attributs, appelés *fG* et *fD*, qui pourront être égaux a *null*.

La *taille* d'un arbre est son nombre de noeuds. La *profondeur* d'un noeud dans un arbre est sa distance a la racine. La profondeur d'un arbre est la profondeur maximum de ses noeuds. L

De la meme façon que pour les listes chainées, où une liste était simplement vue comme la donnée d'une cellule de début, un arbre est alors la donnée d'une cellule particulière étant sa racine. A partir de cette racine, on est donc capable d'explorer tous ses descendants, et donc tour l'arbre.

1: **Arbres : Un peu de Combinatoire**

(a) On appelle *arête* un lien noeud-fils. Si un arbre a n noeuds, combien a-t-il d'arêtes ?

(b) Dans un arbre binaire, donner une borne supérieure sur le nombre de noeuds en fonction de la profondeur de l'arbre.

(c) Dans un arbre binaire, donner une relation entre le nombre de noeuds internes et le nombre de feuilles.

2: **Arbres Binaires : Manipuler la structure**

Dessiner l'arbre binaire `a` correspondant la suite d'instructions suivante :

```
Noeud n = Alloc()
n->fD = Alloc()
n->val=2
n->fD -> val = 4
n->fD->fD = Alloc()
n->fD->fD->val=3
Noeud n2=Alloc()
n2->val=0
n2->fG = Alloc()
n2->fG->val = 3
n->fG = n2
Arbre a = n
```

3: **Arbre : fonctions récursives basiques**

(a) Écrire les fonctions qui renvoient respectivement la taille, le nombre de feuilles, la profondeur max d'un arbre.

(b) Écrire une fonction qui renvoie le noeud qui contient le maximum des valeurs d'un arbre.

(c) Écrire une fonction qui teste si l'arbre passé en paramètre est un chemin, c'est à dire un arbre où chaque noeud possède au plus un fils.

(d) Un peigne gauche est un arbre binaire ou le fils droit de chaque noeud est une feuille. Écrire une fonction qui teste ceci.

4: **Arbres : parcours**

Il existe plusieurs façons de parcourir un arbre. On rappelle ici trois parcours les plus classiques :

— Parcours **préfixe** :

1. on visite la racine
2. on visite le sous-arbre gauche en ordre préfixe
3. on visite le sous-arbre droit en ordre préfixe

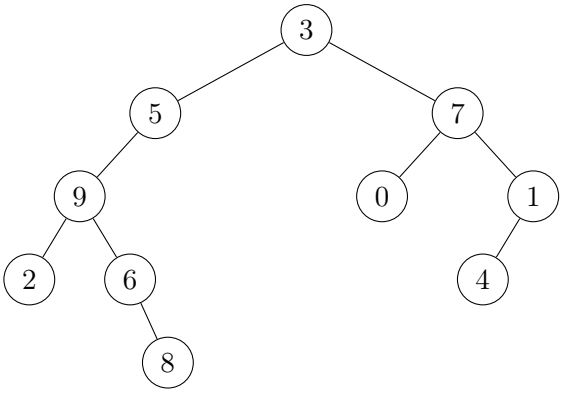
— Parcours **infixe** :

1. on visite le sous-arbre gauche en ordre infixe
2. on visite la racine
3. on visite le sous-arbre droit en ordre infixe

— Parcours **postfixe** :

1. on visite le sous-arbre gauche en ordre postfixe
2. on visite le sous-arbre droit en ordre postfixe
3. on visite la racine

(a) Si au moment où on visite un noeud, on affiche sa valeur, on obtient un affichage préfixe, infixe et postfixe. Donner ces affichages pour l'arbre suivant ?



(b) Un parcours dit en profondeur est un parcours dans lequel on visite successivement les noeuds avec la règle suivante : on doit visiter toujours le noeud le plus éloigné possible de la racine, à condition que ce soit le fils d'un noeud deja visité. Lequel des trois parcours plus haut est un parcours en profondeur ?

(c) Pour écrire des algorithmes itératifs permettant d'effectuer ces parcours, on utilise une structure de données de type PILE (ou LIFO pour "Last In First Out"). Écrire de cette façon un algorithme réalisant un affichage prefixe.

(d) Écrire un algorithme permettant de calculer de façon itérative le nombre de feuilles.

(e) Supposons qu'on remplace la PILE par une FILE (ou FIFO pour "First In First Out") dans l'implémentation du parcours préfixe. Quel affichage obtient-on dans l'arbre donné en exemple au (a) ? Si un arbre quelconque, quel est la propriété du parcours obtenu ? Ce type de parcours peut-il être obtenu de façon récursive ?