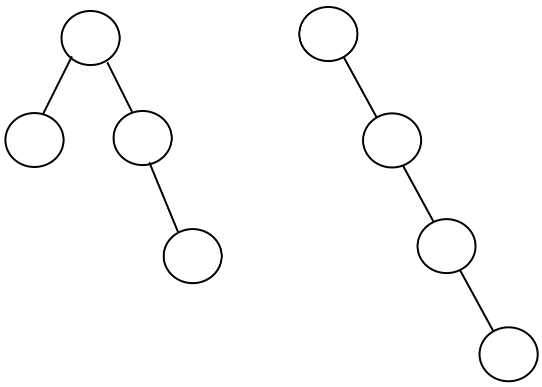
complexité
Examen

Le cheminement interne d'un arbre binaire ordonné, est la somme pour tous les nœuds *n* de l'arbre du nombre de nœuds du chemin allant de la racine au nœud *n*.

Les arbres ci-contre ont des cheminement interne de 8 et de 10 respectivement.



Les arbres binaires ayant le plus petit cheminement interne pour un nombre de nœuds donnés sont les arbres complets et les arbres ayant le plus grand cheminement interne sont les arbres dégénérés. Nous nous proposons de calculer le cheminement interne des arbres dégénérés, puis des arbre parfaits (arbre binaire complet dont le dernier niveau est complètement rempli).

Question 1.

Justifier puis résoudre la récurrence suivante qui donne le cheminement interne pour un arbre dégénéré de *n* nœuds. $\left| \begin{array}{l} t_0 = 0 \\ t_n = t_{n-1} + n \end{array} \right.$

Question 2.

Justifier puis résoudre la récurrence suivante qui donne le cheminement interne pour un arbre parfait de hauteur *h*. $\left| \begin{array}{l} t_1 = 1 \\ t_h = t_{h-1} + h \times 2^{h-1} \end{array} \right.$

Question 3.

Une grammaire des expressions arithmétiques simplifiées est :

```
exp → e ';'
e → t '+' e | t
t → f '*' t | f
f → a |  '(' e ')'
```

Cette grammaire traduite en prolog en utilisant la méthode de la différence de listes donne la suite de règles :

```
exp(L, R) :- e(L, [(;)|R]).
e(L, R) :- t(L, [+|R1]), e(R1, R).
e(L, R) :- t(L, R).
t(L, R) :- f(L, [*|R1]), t(R1, R).
t(L, R) :- f(L, R).
f([a|R],R).
f(['('|R1], R):- e(R1, [')'|R]).
```

Donner la récurrence qui exprime le nombre de fois que le prédicat `e` est évalué lorsqu'on analyse des expressions arithmétiques constituées de n niveaux de parenthèses imbriquées `(((((a))))) ;` où `a` est imbriqué dans `n` niveaux de parenthèses. Résoudre cette récurrence.