Second concours de l'ÉNS de Lyon
Épreuve d'informatique

ÉNS de Lyon

45 minutes – 11 juillet 2007

Avertissement et conseils

- *Les questions sont données en italique.*
- *Les algorithmes seront décrits dans un langage au choix du candidat.*
- *L'énoncé proposé n'est pas nécessairement conçu pour être totalement résolu dans le temps imparti. Le candidat cherchera en priorité à élaborer une réponse de qualité à une ou plusieurs des questions. Il sera tenu compte de la rigueur des réponses dans l'évaluation de l'épreuve.*
- *L'exercice pourra déboucher sur des questions supplémentaires posées au candidat après la fin de la présentation des réponses préparées.*

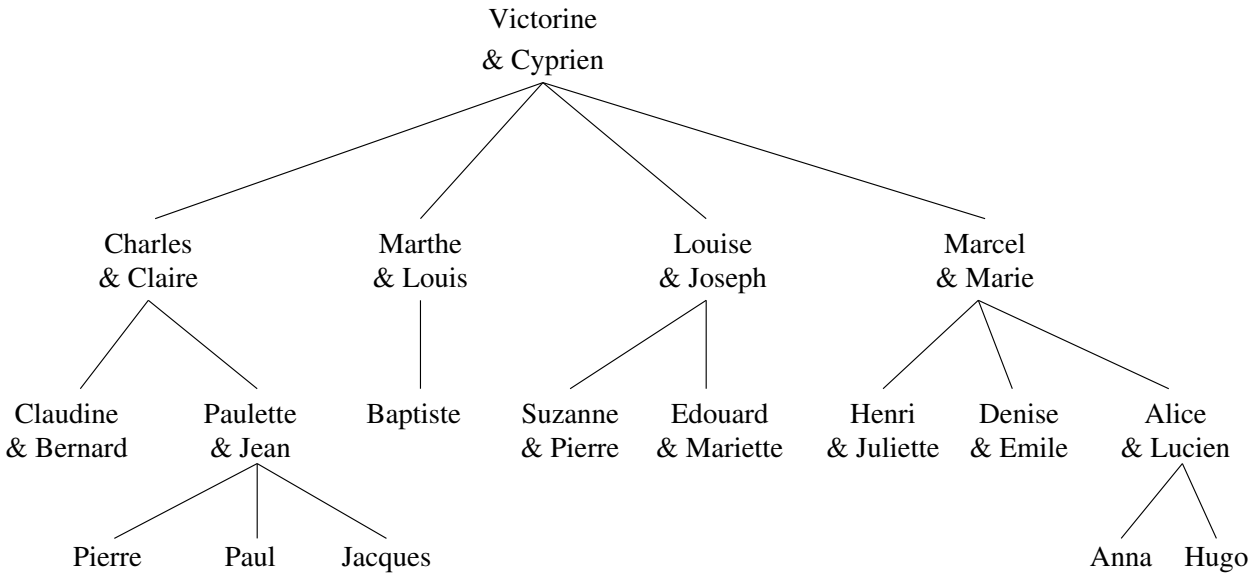
Comparer des arbres

On s'intéresse dans cet énoncé à la comparaison de deux arbres quelconques : sont-ils égaux ou différents ?

1 Arbres

Définition : arbre.
Un arbre est :
- soit une constante dont la valeur est une chaîne de caractères, on dit alors qu'il s'agit d'une feuille ;
- soit un nœud : un couple formé d'une valeur (une chaîne de caractères) et d'une liste finie et ordonnée de sous-arbres appelés ses enfants.

Par exemple un arbre généalogique est un arbre :



Par convention, on notera `null` le pointeur sur rien.

Question 1. *Définissez un type de données pour les listes d'enfants, que vous appellerez* `type_enfants`*. Définissez un type de données pour les arbres, que vous appellerez* `type_arbre`*.*

Remarque. Notre définition et notre type de données n'interdisent pas de construire des arbres infinis, c'est-à-dire qui contiennent des nœuds qui sont leurs propres descendants. On suppose que cela n'arrivera jamais dans les arbres que nous aurons à manipuler et il n'est donc pas demandé de le vérifier.

Définition : hauteur d'un nœud d'un arbre.
La hauteur d'un nœud d'un arbre est :
– 1 s'il s'agit d'une constante (feuille) ;
– le maximum des hauteurs de ses enfants plus 1 pour les nœuds qui ne sont pas des feuilles.

Question 2. *Écrivez un algorithme qui prend en entrée un arbre et qui calcule sa hauteur.*

2 Comparaison et unification d'arbres

2.1 Deux arbres sont-ils égaux ?

Soient deux arbres A_1 et A_2 donnés, on cherche à savoir s'ils sont égaux.

Question 3. *Écrivez un algorithme qui teste si $A_1 = A_2$ et qui retourne True si A_1 et A_2 sont égaux, False sinon.*
Prouvez la terminaison de votre algorithme.

2.2 Deux termes sont-ils identiques ?

On se place maintenant dans le cas plus général où certains sous-arbres sont des variables, une variable pouvant représenter tout un sous-arbre.

Définition : terme.
Un terme est
- soit une constante dont la valeur est une chaîne de caractères, on dit alors qu'il s'agit d'une feuille ;
- soit une variable (que l'on écrira en majuscules) ;
- soit un nœud : un couple formé d'une valeur (une chaîne de caractères) et d'une liste finie et ordonnée de sous-termes appelés ses enfants.

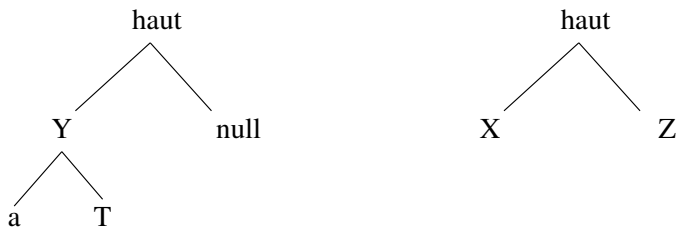
On se demande si deux termes sont identiques, c'est-à-dire, dans le cas où ils ne contiennent pas de variables, s'ils sont égaux, et dans le cas contraire où ils contiennent des variables, s'il existe une affectation de ces variables telles que les deux arbres obtenus en substituant ces variables par leurs valeurs sont égaux.

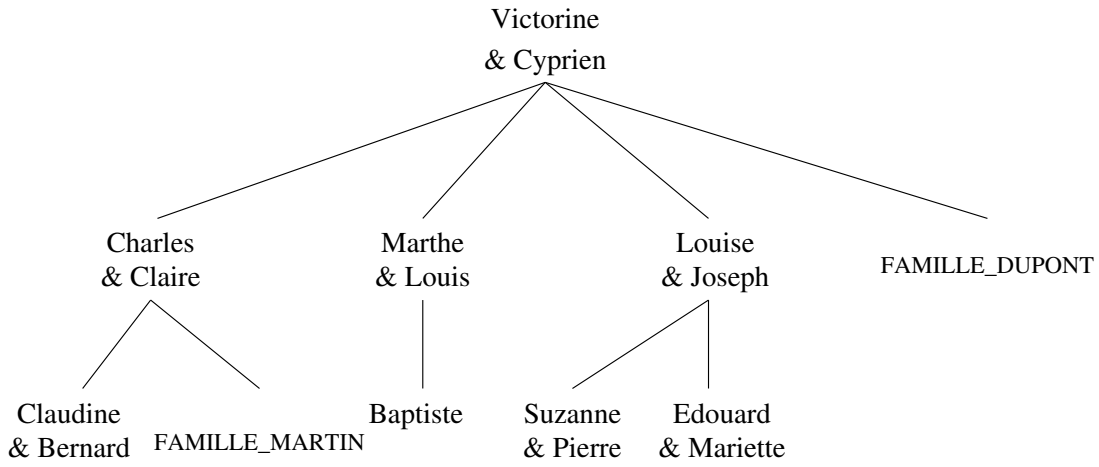
Par exemple les deux termes représentés ci-après sont identiques, il suffit d'affecter à la variable `FAMILLE_MARTIN` le sous-arbre correspondant à la famille fondée par Paulette&Jean, à la variable `FAMILLE_DURAND` le sous-arbre correspondant à la famille fondée par Louise&Joseph et d'affecter l'une à l'autre les variables `FAMILLE_DUPONT` et `FAMILLE_D`.

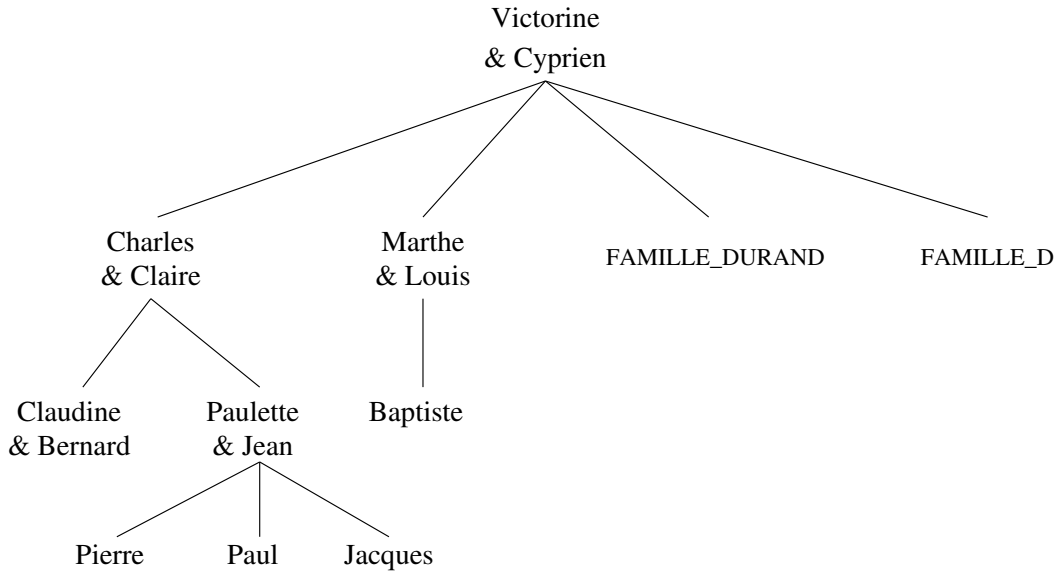
Question 4. *Modifiez les types de données définis pour représenter les arbres dans la partie précédente, afin de pouvoir représenter les termes.*

On suppose que l'on dispose de deux procédures :
- `est_descendant_de (x : type_terme, y : type_terme)` qui retourne True si `x` est un sous-terme de `y` et False sinon ;
- `substitue(t : terme, x : terme, y : terme)` qui substitue dans le terme `t` le sous-terme `x` par le terme `y` : au départ `x` est un sous-terme de `t`, à l'arrivée `y` est un sous-terme de `t`.

Question 5. *Écrivez un algorithme qui teste si deux termes donnés T_1 et T_2 sont identiques.*
Démontrez que votre algorithme termine.
Faites-le tourner à la main sur l'exemple ci-dessous :




Victorine
& Cyprien
Charles
& Claire
Marthe
& Louis
Louise
& Joseph
FAMILLE_DUPONT
Claudine
& Bernard
FAMILLE_MARTIN
Baptiste
Suzanne
& Pierre
Edouard
& Mariette


Victorine
& Cyprien
Charles
& Claire
Marthe
& Louis
FAMILLE_DURAND
FAMILLE_D
Claudine
& Bernard
Paulette
& Jean
Baptiste
Pierre
Paul
Jacques