Programmation TP N 7

Arbre lexicographique.

Définition.

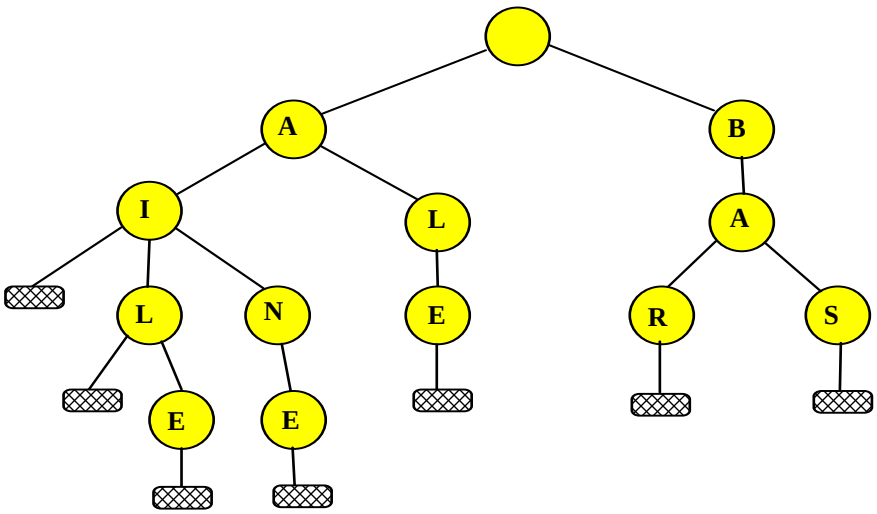
Nous nous proposons de représenter une liste de mots ordonnée. Si l'ordre sur les mots est obtenu comme un ordre lexicographique à partir des composants des mots, nous pouvons représenter un ensemble de mots en utilisant la structure d' « arbre de recherche lexicographique ».

Exemple :

Soit le dictionnaire constitué des mots suivants :

AI
AIL
AILE
AINE
ALE
BAR
BAS

Il peut être représenté par l'arbre suivant :



Chaque mot du dictionnaire est associé à un chemin de la racine à un noeud  dans l'arbre équivalent précédent.

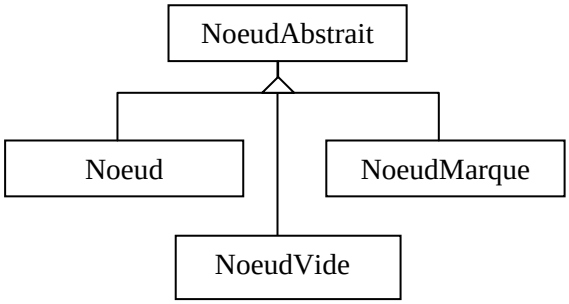
Implantation.

Nous représentons l'arbre lexicographique sous la forme d'un arbre formé de nœuds *Noeud* et *NoeudMarque*.

```
class NoeudA {
   private NoeudA alt;
   public NoeudA(NoeudA a) {…}
}
```

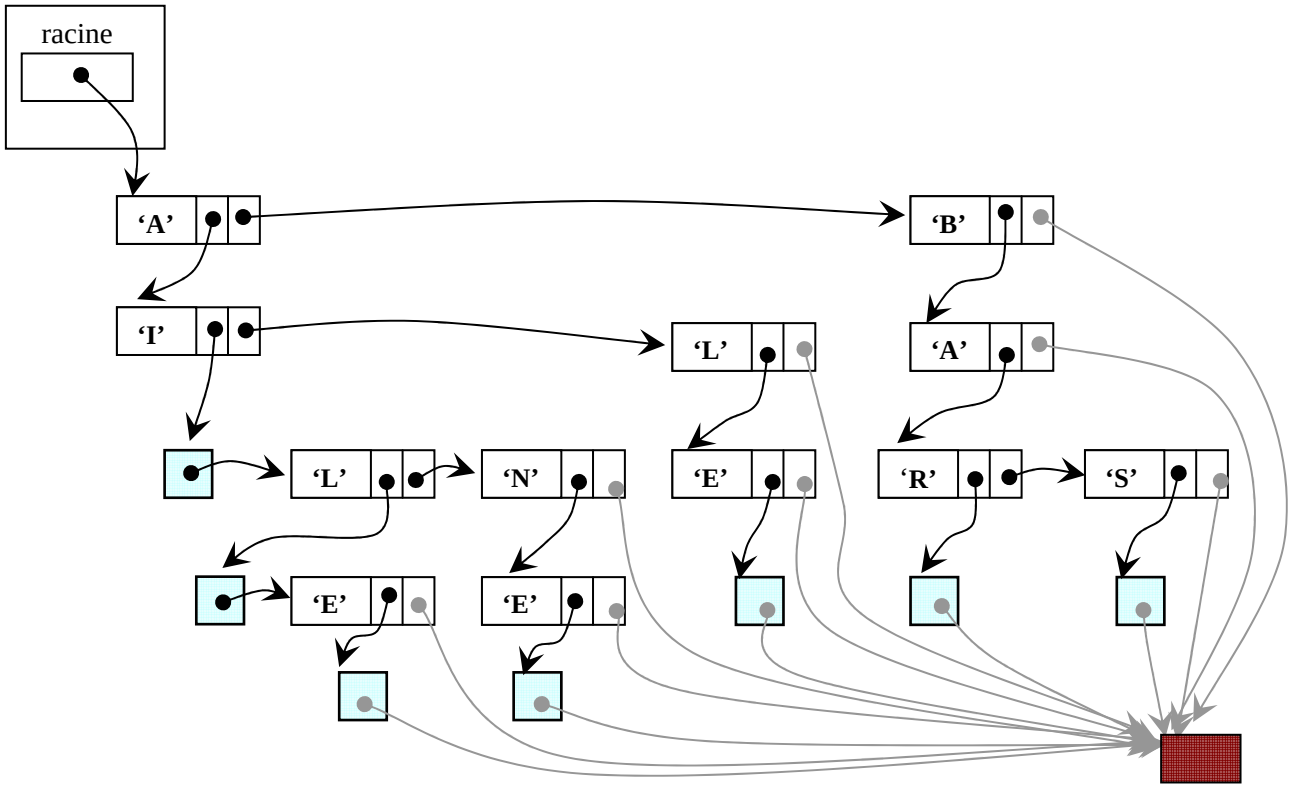
```
class Noeud
      extends NoeudA {
   private char info;
   private NoeudA succ;
   public Noeud(char c,
                NoeudA s, NoeudA a) {…}
}
```

```
class NoeudMarque
      extends NoeudA {
   public NoeudMarque (NoeudA a) {…}
}
```




```
class NoeudVide
      extends NoeudA {
   static NoeudVide s=null ;
   protected NoeudVide () {…}
   public NoeudVide getInstance(){…}
}
```

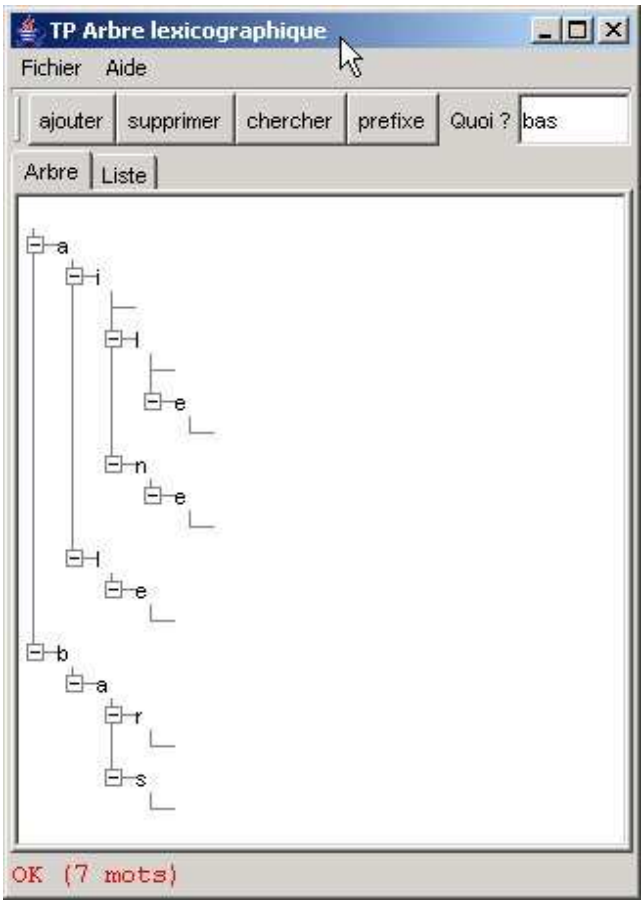
L'arbre précédent est donc représenté par :



Ecrire la définition d'une classe **ArbreLexicographique** munie des méthodes :

- Le(s) constructeur(s).
- Une méthode ***boolean appartient(String s)*** qui permet de savoir si un mot appartient ou non à l'arbre.
- Une méthode ***boolean prefixe(String s)*** qui permet de savoir si un mot est un préfixe d'un ou plusieurs mots se trouvant dans le dictionnaire.
- Une méthode ***boolean ajout(String s)*** qui ajoute un mot dans l'arbre. Si le mot est déjà présent dans l'arbre, cette fonction ne doit rien faire et retourne *false*, sinon elle retourne *true*.
- Une méthode ***boolean suppression(String s)*** qui supprime un mot de l'arbre. Si ce mot est absent du dictionnaire cette méthode ne doit rien faire et retourne *false*, sinon elle retourne *true* après avoir oté le mot de l'arbre.
- Une méthode ***String toString()*** qui construit une chaîne de caractères contenant tous les mots de l'arbre, séparés par des retour à la ligne.
- Les méthodes ***void sauve(String nomFichier)*** et ***void restaure(String nomFichier)*** qui permettent de sauvegarder et restaurer un arbre sur un fichier disque.
- Une méthode ***int nbMots()*** qui permet d'obtenir le nombre de mots présents dans l'arbre.

L'interface utilisateur aura l'allure suivante :



Pour construire l'interface utiliser :

- *JTabbedPane* http://prevert.upmf-grenoble.fr/Prog/Java/swing/JTabbedPane.html
- *JTree* http://prevert.upmf-grenoble.fr/Prog/Java/swing/JTree.html

Le menu *Fichier* contient les trois items *Sauver*, *Restaurer* et *Quitter*.