Feuille d'exercices n°2 de l'option d'informatique.

CAML et les arbres

1. Étant donné un type arbre binaire donné par la définition CAML suivante :

```
type arbre =
nœud of arbre * arbre
| feuille;;
```

Écrire les fonctions suivantes :

- **hauteur** qui calcule la hauteur (ou la profondeur) d'un arbre,
- **nbnœuds** qui calcule le nombre de nœuds d'un arbre,
- **nbfeuilles** qui calcule le nombre de feuilles d'un arbre.

2. Soit le type d'arbre binaire suivant, contenant des chaînes de caractères à chaque nœud,

```
type dico =
nœud of dico * string * dico
| vide;;
```

Écrire les fonctions suivantes :

- **creearbre mot** qui crée un arbre contenant **mot** et des fils gauche et droit vides ,
- **insere mot** qui ajoute un mot dans l'arbre dictionnaire s'il n'y est pas. (lors de l'insertion, on respecte l'ordre lexicographique, c'est à dire que si le mot à insérer est avant le mot du nœud dans l'ordre lexicographique, on l'insère dans son sous arbre gauche sinon on l'insère dans le sous arbre droit).
- **lit arbre_dico** qui parcourt l'arbre de façon à afficher les mots du dictionnaire dans l'ordre lexicographique,
- **liste_de_arbre** qui transforme un arbre du dictionnaire en une liste de mots rangés dans l'ordre lexicographique,
- **arbre_de_liste** qui transforme une liste de mots en l'arbre du dictionnaire associé,
- **supprime mot** qui supprime un mot dans l'arbre dictionnaire tout en respectant la structure d'arbre dictionnaire,

3. On considère les opérateurs logiques "et", "non", "ou", "impliq", "equiv", "xou", "nonet", ainsi que les constantes "vrai" et "faux" définis en CAML par :

```
type opelog = et | non | ou | impliq | equiv | xou | nonet;;
```

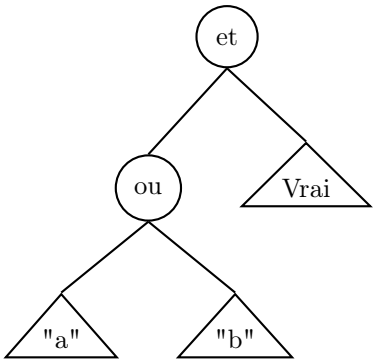
type constlog = vrai | faux;;

- Définir un type d'arbre convenant à la représentation des expressions logiques dont les nœuds sont des opérateurs logiques et les feuilles, des constantes ou des noms de variables.

- Écrire une fonction **arbexpr** qui transforme une liste de chaînes de caractères représentant une expression logique bien parenthésée ne contenant que des opérateurs logiques précités, des constantes et des variables en l'arbre logique correspond,

  par exemple, l'expression ((a ou b) et vrai) sera représentée par la liste :

  ["(";"(";"a";"ou";"b";")";"et";"vrai";")"]et la fonction **arbexpr** retournera l'arbre suivant :



- Écrire une fonction qui transforme un arbre logique ne contenant que des opérateurs logiques et des constantes en un arbre logique ne contenant que des constantes et les opérateurs logiques "et", "ou" et "non" correspondant,

- Écrire une fonction **evalexpr** qui évalue un arbre logique, c'est à dire donne sa valeur vrai ou faux (si l'arbre contient une variable la fonction **evalexpr** doit retourner un message d'erreur).

- Écrire une fonction qui retourne la liste des noms de variables contenus dans un arbre logique,

- Écrire une fonction qui vérifie si un arbre logique est une tautologie ou non.