TP : Dictionnaire via un arbre.

Les mots d'un dictionnaire peuvent être représentés par un arbre en faisant en sorte que les préfixes communs à plusieurs mots apparaissent une seule fois :

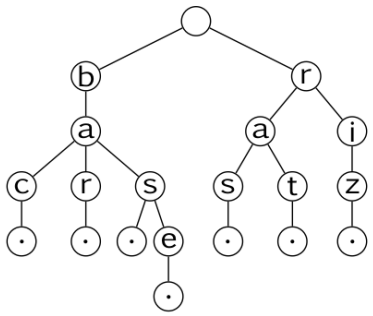

FIGURE 1: Les mots représentés sont : bac, bar, bas, base, ras, rat, riz.

Comme on le voit :

— la racine ne comporte pas d'étiquette ;

— un nœud peut avoir un nombre quelconque de fils ;

— on fait terminer chaque mot par un point : c'est utile pour faire figurer deux mots dont l'un est préfixe de l'autre dans le dictionnaire (comme « bas » et « base »).

On pourrait représenter un tel arbre comme un « arbre général » comme dans le cours, néanmoins il existe une autre possibilité. Un arbre général A peut se représenter comme un arbre binaire B, via la représentation « fils gauche, frère droite » :

— à chaque nœud de A est associé un nœud de B ;

— le sous-arbre gauche d'un nœud de B est associé à la descendance du nœud correspondant dans A ; en particulier, si un nœud de A possède au moins un fils, celui le plus à gauche sera le fils gauche du nœud dans B.

— le sous-arbre droit d'un nœud de B est associé aux frères droits du nœud correspondant dans A ; en particulier si un nœud de A possède un frère droit, il sera le fils droit du nœud correspondant dans B.

La racine d'un arbre B associé à un arbre A quelconque ne possède pas de fils droit puisque la racine dans A n'a pas de frère droit. Ici, la racine étant inutile, on représentera directement l'arbre par le sous-arbre gauche de l'arbre binaire associé. Dans l'exemple, l'étiquette de la racine de l'arbre binaire sera donc b, le fils gauche de la racine sera d'étiquette a, le fils droit d'étiquette r)

Question 1. Dessiner soigneusement l'arbre binaire représentant le dictionnaire de l'exemple.

On représente un tel arbre binaire via le type suivant.

```
type dict = V | N of char * dict * dict ;;
```

Rappel sur les chaînes de caractères. `s` désigne une chaîne de caractères (type `string`)

— `String.length s` : longueur de la chaîne ;

— `s.[i]` accès au i-ème caractère, pour $0 \leq i <$ `String.length s`. Un élément d'une chaîne de caractères est de type `char`.

— `String.sub s i t` extrait de `s` la sous-chaîne démarrant à l'indice `i` de taille `t`.

Une chaîne de caractères est immuable en Ocaml (on ne peut modifier un caractère). Le type `bytes` (chaîne d'octets) est un type mutable qui permet de créer une chaîne de caractères par modification.

— `Bytes.make : int -> char -> bytes` crée un objet de type `bytes` d'une certaine longueur.

— `Bytes.of_string s` : crée un objet de type `bytes` à partir d'une chaîne `s`.

— `Bytes.set : bytes -> int -> char -> unit` permet de modifier un élément d'un `bytes`.

— `Bytes.to_string` : obtenir une chaîne à partir d'un `bytes`.

Question 2. Écrire un code permettant de représenter l'arbre ci-dessus en Caml dans la variable `a`. N'hésitez pas à procéder par étapes !

Question 3. Rédiger une fonction `chercher : string -> dict -> bool` qui détermine si un mot appartient à un dictionnaire donné. Introduire une fonction auxiliaire !

```
# chercher "bras" a, chercher "bas" a, chercher "base" a ;;
- : bool * bool * bool = (false, true, true)
# chercher "ri" a, chercher "riz" a, chercher "rize" a ;;
- : bool * bool * bool = (false, true, false)
```

Question 4. Rédiger une fonction `branche : string -> dict` qui prend pour argument un mot `s` et retourne pour résultat l'arbre réduit à la branche qui code le mot `m`. La racine sera la première lettre de `s`.

```
# branche "abc" ;;
- : dict = N ('a', N ('b', N ('c', N ('.', V, V), V), V), V)
```

Question 5. Définir une fonction `inserer : string -> dict -> dict` qui prend pour arguments un mot `m` et un arbre `a` et qui retourne l'arbre auquel le mot `m` a été ajouté, s'il n'y était pas déja. On pourra utiliser `String.sub`.

Question 6. Rédiger une fonction `creer : string list -> dict` qui prend pour argument une liste de mots et qui retourne le dictionnaire créé à l'aide de ces mots.

Question 7. Rédiger enfin une fonction `extraire : dict -> string list`. On utilisera le type `Bytes`.