Feuille d'exercices $n^{\circ}5$

Exercice 1. Les fonctions de cet exercice prennent en paramètres plusieurs listes qu'on suppose de même longueur. Si ce n'est pas le cas, une erreur devra être signalée par votre fonction.

1. Donner le type et écrire une fonction polymorphe map2 récursive terminale qui prend en paramètres une fonction à deux arguments et deux listes de même longueur et qui applique cette fonction aux éléments des deux listes. Exemple :

```
1 # map2 (+) [1;2;3] [3;2;1] ;;
2 - : int list = [4;4;4]
```

- 2. Donner le type et écrire une fonction polymorphe select qui prend en paramètres une fonction filter de type 'a -> bool et trois listes 11, 12 et 13 de même longueur (respectivement de type 'a list, 'b list et 'b list) et qui renvoie une liste dont le ieme élément est
 - le ieme de 12 si le résultat de filter appliquée au ieme de 11 est true
 - le ieme de 13 sinon

```
1 # select (fun x -> x=0) [1;0;0;1] ['a';'b';'c';'d'] ['A';'B';'C';'D'] ;;
2 - : char list = ['A';'b';'c';'D']
```

3. En considérant la fonction f définie ci-dessous, quel est le résultat de l'évaluation de l'expression donnée ?

```
1 let f = fun filter 11 12 13 ->
2 select filter 11 (select filter 11 12 13) (select filter 11 13 12)
3 # f (fun x -> x=0) [1;0;0;1] ['a';'b';'c';'d'] ['A';'B';'C';'D'];;
4 -: ??????
```

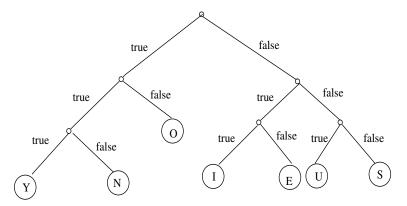
En combinant les fonctions map2 et select, écrire une fonction polymorphe min qui prend en paramètres deux listes et qui renvoie une liste dont le ieme élément est le minimum des iemes éléments de ces deux paramètres (N.B. vous ne devez pas définir de nouvelle fonction récursive) :

```
1 # min [1;5;3;8] [2;6;7;6]
2 -: int list = [1;5;3;6]
```

Exercice 2. Arbres binaires feuilles

Un arbre binaire feuille (abf) est un arbre binaire dont seules les feuilles contiennent des informations.

- 1. Définir un type polymorphe abf.
- 2. Une application des abf est le codage/décodage. Étant donnés un abf et un code représenté par une liste de booléens, on obtient l'information codée en parcourant en même temps l'arbre et la liste : si la tête de liste vaut true, on va à gauche dans l'arbre sinon, on va à droite. Si l'arbre et la liste sont bien formés, on arrive sur une feuille lorsque la liste est vide. L'information cherchée s'y trouve. Par exemple, pour l'arbre a



La liste [true; true; false] code N I est codé par [false; true; true] [true; false; true] est un mauvais code

A n'a pas de code.

Définir l'arbre a ci-dessus en fonction du type abf que vous avez défini.

- 3. Écrire une fonction decode_info qui prend un abf et un code et qui retourne l'information correspondante ou lève une exception.
- 4. Écrire une fonction code_info qui prend un abf et une information et qui retourne le code correspondant ou lève une exception.
- 5. Écrire une fonction build_dictionnaire qui prend un abf en paramètre et qui renvoie une liste formée de toutes les informations et de leurs codes associés.
- 6. Écrire une fonction cherche_dictionnaire qui prend en arguments un dictionnaire et une information et qui renvoie le code de l'information. (Penser aux listes d'association).