Exercices d'été

Exercice 1.
Ecrire une fonction `liste_premiers` qui à partir d'un entier n renvoie la liste de ses facteurs premiers.
Par exemple `liste_premiers 315` doit renvoyer `[3;3;5;7]`.

Exercice 2.

1. Ecrire une fonction `compress` qui à partir d'une liste renvoie la liste obtenue après suppression des répétitions d'éléments identiques : par exemple

   `compress ["a";"a;"b";"b";"b";"a";"c";"c"]` doit renvoyer `["a";"b";"a";"c"]`.

2. Ecrire une fonction `compress_tout` qui à partir d'une liste renvoie une liste d'éléments distincts deux à deux contenant tous les éléments de la liste initiale.

Exercice 3.

1. Ecrire une fonction `split` qui à partir d'une liste $[a_0;\ldots;a_{n-1}]$ et d'un entier k renvoie les deux listes $[a_0;\cdots;a_{k-1}]$ et $[a_k;\cdots;a_{n-1}]$.

2. Ecrire une fonction `rotate` qui à partir d'une liste `li` et d'un entier k, renvoie une liste obtenue par rotation de k places par la gauche :

   par exemple `rotate ["a";"b";"c";"d";"e"] 2` doit renvoyer `["c";"d";"e";"a";"b"]`.

Exercice 4.
On utilise le type suivant pour implémenter la structure d'arbre binaire :
```
type 'a arbre = Vide | Noeud of 'a * 'a arbre * 'a arbre;;
```

1. Ecrire une fonction `est_miroir` qui à partir de deux arbres binaires, renvoie `true` si un arbre est le miroir de l'autre (sans prendre en compte les étiquettes des noeuds), et `false` sinon.

2. Ecrire une fonction `est_symetrique` qui à partir d'un arbre renvoie `true` si l'arbre est symétrique par rapport à la ligne verticale passant par la racine.

Exercice 5.
On utilise le type `arbre` suivant :
```
type 'a arbre = Vide | Noeud of 'a * 'a arbre * 'a arbre;;
```
Ecrire une fonction `noeuds_meme_hauteur` qui à partir d'un arbre et un entier k, renvoie la liste des noeuds situés à la hauteur k dans l'arbre.

Exercice 6. (facultatif)
Parties II et III du sujet Centrale 2023 :
https ://www.concours-centrale-supelec.fr/CentraleSupelec/2023/MP/I014.pdf