TD 3

Liste et récursivité

Exercice 1 :

1. Écrire une fonction `der` qui renvoie le dernier terme d'une série.

   ```
   val der : 'a list -> 'a = <fun>
   ```

2. Écrire une fonction `avant_der` qui renvoie l'avant dernier terme d'une série.

   ```
   val avant_der : 'a list -> 'a = <fun>
   ```

3. Écrire une fonction `nieme` qui renvoie le nième terme d'une série pour un n donné.

   ```
   val nieme : 'a list -> int -> 'a = <fun>
   ```

Exercice 2 : Écrire la fonction `minimax` qui donne le plus grand et le plus petit élément d'une liste :

```
val minimax : 'a list -> 'a * 'a = <fun>
```

Exercice 3 : Écrire la fonction `cat` qui concatène deux listes.

```
val cat : 'a list -> 'a list -> 'a list = <fun>
```

Exercice 4 : Écrire la fonction `map_liste` qui applique une fonction à tous les éléments d'une liste, et renvoie la liste modifiée.

```
val map_liste : ('a -> 'b) -> 'a list -> 'b list = <fun>
map_liste (fun x -> ( x > 5) ) [7;4;3;1;9;7;4;6];;
- : bool list = [true; false; false; false; true; true; false; true]
```

Exercice 5 : Écrire une fonction `occurrence` tel que `occurrence element liste` retourne la liste des positions des occurrences de élément dans liste.

```
val occurrence : 'a -> 'a list -> int list = <fun>
occurrence "a" [ "a";"bb";"a";"jj";"n";"a"];;
- : int list = [1; 3; 6]
```

Exercice 6 : Soit $n \in \mathbb{N}^*$ et $l = (x_i)_{1 \leq i \leq n}$ une liste de n éléments.
Un palier de l est une liste d'éléments consécutifs égaux , et de longueur maximale pour cette propriété.
Ainsi, la liste $[2;2;8;3;3;3]$ comporte trois paliers : $[2;2]$, $[8]$, $[3;3;3]$.
On peut donner une définition par induction structurelle :

- La liste vide ne comporte aucun palier.
- Soit $l = (x_i)_{1 \leq i \leq n}$ une liste non vide ; notons j le plus grand indice tel que $x_i = x_1$ pour tout $i \in [\![1;j]\!]$: la liste $(x_i)_{1 \leq i \leq j}$ est le premier palier de l ; si $j = n$, c'est le seul ; sinon les autres paliers sont ceux de $(x_i)_{j < i \leq n}$.

1. Écrire une fonction `palier_tete` qui renvoie le couple formé du premier palier et la reste de la liste.

   ```
   val palier_tete : 'a list -> 'a list * 'a list = <fun>
   ```

2. Écrire une fonction `palier` qui renvoie la liste des paliers de la liste entrée.

   ```
   val palier : 'a list -> 'a list list = <fun>
   ```

Exercice 7 : **Tri par sélection :**

Le tri par sélection (ou tri par extraction) est un des algorithmes de tri les plus triviaux. Il consiste en la recherche soit du plus grand élément (ou le plus petit) que l'on va replacer à sa position finale c'est-à-dire en dernière position (ou en première), puis on recherche le second plus grand élément (ou le second plus petit) que l'on va replacer également à sa position finale c'est-à-dire en avant-dernière position (ou en seconde), etc., jusqu'à ce que le tableau, ou la liste, soit entièrement trié.

1. Écrire la fonction `min_list` qui renvoie pour une liste donnée l le minimum de la liste ainsi que le reste de la liste.

   ```
   val min_list : 'a list -> 'a * 'a list = <fun>
   ```

2. Écrire la fonction `tri_selec` qui renvoie la liste triée par sélection :

   ```
   val tri_selec : 'a list -> 'a list = <fun>
   ```