

# Programmation Fonctionnelle : TD4

Université de Tours

Département informatique de Blois

*Fonctions de tri*

\*  
\* \*

## Appropriation du cours

1. On reprend le code suivant qui permet la réalisation du tri par insertion sur des listes d'entiers.

```
let rec insere x l = match l with
  [] -> [x]
| h::t -> if (h > x) then x::l
          else h::(insere x t);;

let rec tri_insertion l = match l with
  [] -> []
| [x] -> [x]
| h::t -> insere h (tri_insertion t);;
```

Exécuter cet algorithme pour la liste  $l = [0; -1; 1; -2; 2; -3; 3]$  et  $l' = [13; 11; 7; 5; 3; 2]$ .

Suivez l'exécution à l'aide de la commande `#trace`.

2. On reprend le code suivant qui permet la réalisation du tri par fusion sur des listes d'entiers.

```
let rec separe l = match l with
  [] -> ([], [])
| [x] -> ([x], [])
| a::b::c -> let (l1, l2) = separe c in
             (a::l1, b::l2);;

let rec fusionne l1 l2 = match (l1, l2) with
  ([], l2) -> l2
| (l1, []) -> l1
| (a::b, h::t) -> if (a <= b) then a::(fusionne b l2)
                  else h::(fusionne l1 t);;

let rec tri_fusion l = match l with
  [] -> []
| [x] -> [x]
| h::t -> let (l1,l2) = (separe l) in
          fusionne (tri_fusion l1) (tri_fusion l2);;
```

Exécuter cet algorithme sur les précédents exemples et avec la commande `#trace`.

## Problème 1

1. Écrire la spécification et l'algorithme d'une fonction **partitionne** 1 p qui prend en paramètre une liste d'entiers  $l$  et un entier  $p$  et retourne pour résultat un couple de listes  $(l_1, l_2)$  tel que  $l = l_1 \cup l_2$  où :

$$l_1 = \bigcup_{e \in l | e \leq p} e \quad \text{et} \quad l_2 = \bigcup_{e \in l | e > p} e = l \setminus l_1$$

Plus simplement,  $l_1$  est la liste avec tous les éléments  $e$  de  $l$  plus petits ou égaux à  $p$ , et  $l_2$ , la liste de tous les éléments  $e$  de  $l$  strictement plus grands que  $p$ .

2. Écrire la fonction **tri\_rapide** 1 du cours qui prend une liste  $l$  et l'ordonne selon la relation  $\leq$ .

## Problème 2

1. Écrire la spécification et l'algorithme d'une fonction **extract\_min** 1 qui prend en paramètre d'entrée une liste d'entiers  $l$  et retourne pour résultat un couple  $(min, l')$  où :

$$min = \min_{x \in l} \{x\} \quad \text{et} \quad l' = l \setminus \{min\}$$

Plus simplement,  $min$  est le plus petit élément de  $l$  et  $l'$  est la liste  $l$  sans l'élément  $min$ .

On pourra créer plusieurs fonctions intermédiaires afin de simplifier l'algorithme principal.

2. Écrire la fonction **tri\_selection** 1 du cours qui prend une liste  $l$  et l'ordonne selon la relation  $\leq$ .