Corrigé

Tri d'une liste

Exercice 1. tri par insertion

```
let rec insere x = function
  | []              -> [x]
  | t::q when t < x -> t::(insere x q)
  | l               -> x::l ;;

let rec insertion_sort = function
  | []   -> []
  | [a]  -> [a]
  | t::q -> insere t (insertion_sort q) ;;
```

Exercice 2. tri fusion

```
let rec split = function
  | []      -> [], []
  | [a]     -> [a], []
  | a::b::q -> let l1, l2 = split q in a::l1, b::l2 ;;

let rec merge l1 l2 = match (l1, l2) with
  | [], _                       -> l2
  | _, []                       -> l1
  | t1::q1, t2::_ when t1 < t2 -> t1::(merge q1 l2)
  | _, t2::q2                   -> t2::(merge l1 q2) ;;

let rec mergesort = function
  | []  -> []
  | [a] -> [a]
  | l   -> let (l1, l2) = split l in
           merge (mergesort l1) (mergesort l2) ;;
```

Exercice 3. tri rapide

```
let rec partition x = function
  | []               -> [], []
  | t::q when t <= x -> let l1, l2 = partition x q in t::l1, l2
  | t::q             -> let l1, l2 = partition x q in l1, t::l2 ;;

let rec quicksort = function
  | []   -> []
  | [a]  -> [a]
  | t::q -> let l1, l2 = partition t q in (quicksort l1) @ (t::(quicksort l2)) ;;
```