```
type intervalle == int * int ;;

(* question 1 *)

let chevauche i j = fst j <= snd i && fst i <= snd j ;;


type int_tree = Nil | Noeud of intervalle * int * int_tree * int_tree ;;


(* question 2 *)

let maxi = function
  | Nil                -> failwith "maxi"
  | Noeud (_, m, _, _) -> m ;;


(* question 3 *)

let rec recherche i = function
  | Nil                                                    -> raise
Not_found
  | Noeud (j, _, _, _) when chevauche i j                  -> j
  | Noeud (_, _, fg, _) when fg <> Nil && fst i <= maxi fg -> recherche i
fg
  | Noeud (_, _, _, fd)                                    -> recherche i
fd ;;


(* question 4 *)

let cons j fg fd =
  let maximum j = fun
    | Nil Nil -> snd j
    | Nil a2  -> max (snd j) (maxi a2)
    | a1 Nil  -> max (snd j) (maxi a1)
    | a1 a2   -> max (snd j) (max (maxi a1) (maxi a2))
  in Noeud (j, maximum j fg fd, fg, fd) ;;


(* question 5 *)

let insere i a =
  let rec partitionne = function
    | Nil -> Nil, Nil
    | Noeud (j, _, fg, fd) when fst j < fst i -> let a1, a2 = partitionne
fd in
                                               cons j fg a1, a2
    | Noeud (j, _, fg, fd)                    -> let a1, a2 = partitionne
fg in
                                               a1, cons j a2 fd
  in
  let fg, fd = partitionne a in
  cons i fg fd ;;


(* question 6 *)

let rec supprime i a =
  let rec supprime_min = function
  | Nil                   -> failwith "supprime_min"
  | Noeud (j, _, Nil, fd) -> j, fd
  | Noeud (j, _, fd, fg)  -> let k, f = supprime_min fd in
                            k, cons j f fd
  in match a with
    | Nil -> Nil
    | Noeud (j, _, Nil, fd) when i = j -> fd
    | Noeud (j, _, fg, Nil) when i = j -> fg
    | Noeud (j, _, fg, fd) when i = j  -> let k, f = supprime_min fd in
                                        cons k fg f
    | Noeud (j, _, fg, fd) when fst i < fst j -> let f = supprime i fg in
                                               cons j f fd
    | Noeud (j, _, fg, fd)                    -> let f = supprime i fd in
                                               cons j fg f ;;
```

```
(* test *)

let a = insere (16, 21) (insere (8, 9) (insere (25, 30) (insere (5, 8)
(insere (15, 23) (insere (17, 19) (insere (26, 26) (insere (0, 3) (insere
(6, 10) (insere (19, 20) Nil))))))))) ;;


recherche (22, 25) a ;;
recherche (11, 14) a ;;
```