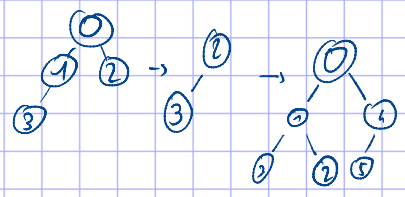
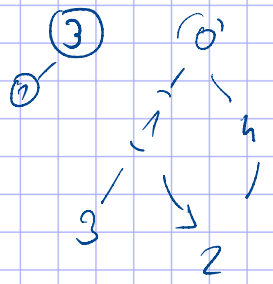
I. Un tas de questions.

1.



2.

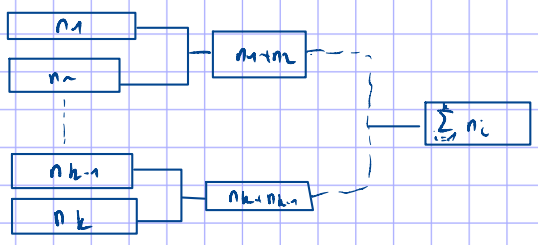


3.
```
let rec is_tas_max heap =
    test = ref true in
    for i = 0 to heap.n - 1 do
        if a.(i) > a.((i-1)/2) then
            test := false
    done;
    !test;;
```

5.
```
let rec fusion l1 l2 = match l1, l2 with
| [], _ -> l2
| _, [] -> l1
| e1::q1, e2::q2 -> if e1 < e2 then
                        e1::fusion q1, l2
                    else
                        e2::fusion l1 q2
```
$\Rightarrow \Theta(n_1+n_2)$

Kfusion :

une étape : $\Theta(\sum n_i)$



k listes k/2 listes ... k/2^p = 1 listes

où $p = \Theta(\log_2(k))$

d'où $\Theta(n \log(k))$.

```
let rec etape ll =
    match ll with
    | l1::l2::q -> fusion (l1, l2) :: etape q
    | _ -> ll
```

```
let rec kfusion ll =
    match ll with
    | [] -> []
    | [l] -> l
    | _ -> kfusion (etape ll).
```

Avec une file de priorité :

```
let kfusion pp :

    let fp = create () in
    List.iter (fun p -> add p fp) pp
    let rec aux () : (* renvoie la fusion de toutes les listes *)
        if is_empty fp then []
        else match extract_min fp with
            |[] -> aux ()
            |e::q -> add q fp; e:: aux() in

    aux().
```