

# UE - Programmation Fonctionnelle

## TD 3.

Exercice 1: type 'a arbre =

| Empty

| Node of bool \* 'a branche list

and 'a branche = 'a \* 'a arbre

↓ réc

type 'a arbre =

'a → element

Node of bool \* 'a branche list

and 'a branche = 'a \* 'a arbre

Exercice 2:

let <sup>reg</sup> appartenir l'character

Quelle est cette structure ?

(Node c b, lb) =

match l'character with

| [] → b

| hd :: tl → match lb with

| [] → false

| (c, a) :: tl →

if c = hd then appartenir <sup>tl</sup> a

else if c > hd then false

else appartenir lc (Node c b, tl)

另一种方法: let rec appartenir lc (Node c b, lb) =

match lc, lb with

| [], - → b

| -, [] → false

| hd :: tl, (c, a) :: tl →

if c = hd then appartenir tl a

else if c > hd then false

else appartenir lc (Node c b, tl)

<fun> recherche: type 'a option =

| None

| Some of 'a

recherche: 'a → ('a \* 'b) list → ('a \* 'b)

recherche 通过参数的值而

```

let rec recherche c lb =
  match lb with
  | [] → None
  | (c₁, a) :: tl → if c₁ = c then Some (c₁, a)
    else if c₁ > c then recherche c tl
    else None

```

<fun> appartient :

bijob ->  
avec "recherche"

```

let rec appartient lc (Node c b lb) =
  match lc with
  | [] → b
  | hd :: tl → match (recherche hd lb) with
    | None → false
    | Some (c₁, a) → appartient lc a

```

<fun> maj :

maj : 'a → 'b → ('a \* 'b) list → ('a \* 'b) list  
意为“更新”  
mise-a-jour

```

let rec maj c nouvelle_b lb =
  match lb with
  | [] → [(c, nouvelle_b)]
  | (c₁, a) :: tl → if c₁ = c then (c, nouvelle_b) :: tl
    else if c₁ < c, then (c, nouvelle_b) :: tl
    else (c, a) :: (maj c nouvelle_b tl)

```