Institut National Universitaire Champollion

Contrôle continu n°2

L1 DLMI
5 décembre 2019

PROGRAMMATION FONCTIONNELLE CAML
Durée : 1h30 - Aucun document autorisé

Consignes :

Vous devez respecter le type des fonctions tel qu'il est donné par l'énoncé ou les exemples.
Vous rendrez à l'issue de cet examen un fichier *nom.prenom.ml* contenant un compte-rendu : les fonctions demandées, leurs types, des exemples d'exécution, vos commentaires.
Le but de ce problème est d'écrire des fonctions permettant la mise à jour d'une liste des membres du personnel d'une entreprise.

1. Listes de noms

On considère dans un premier temps que la liste ne contient que les noms triés des membres du personnel.
Exemple :

```
# let enterprise = ["dupont" ;"durand" ;"martin"] ; ;
enterprise  : string list = ["dupont" ; "durand" ; "martin"]
```

1. Écrire une fonction `nombre : 'a list -> int`, calculant le nombre d'éléments d'une liste.
2. Écrire une fonction `present : 'a * 'a list -> bool`, qui à une chaîne de caractère représentant le nom d'une personne et une liste de noms associe vrai si cette personne est présente dans la liste.
3. Écrire une fonction `embauche : 'a * 'a list -> 'a list` permettant la mise à jour de la liste dans le cas de l'embauche d'un nouveau membre du personnel de nom donné. (La liste résultat doit toujours être triée).

   Exemple :

   ```
   # embauche  ("dupuis",enterprise) ; ;
   -  : string list = ["dupont" ; "dupuis" ; "durand" ; "martin"]
   ```

2. Listes des âges

On considère maintenant que la liste contient des triplets associant à chaque membre du personnel : son nom (une chaîne de caractère), la liste des âges du membre du personnel et de son éventuel conjoint (une liste d'un ou deux entiers) et enfin la liste des âges de ses enfants (une liste d'entiers, éventuellement vide).

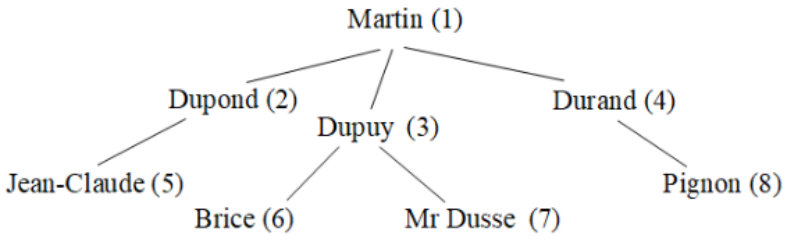
```
Exemple :
# let enterprise = [("dupont",[22 ;25],[3]) ;("durand",[24],[]) ;("martin",[28 ;27],[2 ;4 ;
enterprise : (string * int list * int list) list =
 ["dupont", [22 ; 25], [3] ; "durand", [24], [] ; "martin", [28 ; 27], [2 ; 4 ; 6]]
```

1. Écrire une fonction `nbEnfants : ('a * 'b * 'c list) list -> int` permettant de compter le nombre total d'enfants des membres du personnel (on pourra utiliser la fonction nombre définie précédemment.)
2. Écrire une fonction `nbCelib : ('a * 'b list * 'c list) list -> int` calculant le nombre de personnes isolées (sans conjoint ET sans enfant) parmi les membres du personnel.
3. Écrire une fonction `naissance` permettant la mise à jour de la liste du personnel lors de la naissance d'un enfant dans une famille de nom donné.

```
Exemples :
#NbEnfants(enterprise) ; ;
- : int = 4
     #NbCelib(enterprise) ; ;
-  : int = 1
#naissance ("dupont",enterprise) ; ;
-    :   (string * int list * int list) list = ["dupont", [22 ; 25], [0 ; 3] ; "durand", [
  "martin", [28 ; 27], [2 ; 4 ; 6]]
```

3. Hierarchie

Notre entreprise est une structure fortement hiérarchisée dans laquelle tout le monde veut devenir LE chef. L'organigramme de l'entreprise est actuellement le suivant :



La hiérarchie de l'entreprise peut être modélisée par une liste de triplets (nom,numéro,numéro du chef) avec la convention que LE chef (la racine de l'arborescence) est son propre chef, c'est ça un chef! Ainsi l'entreprise peut être définie par :

```
#let enterprise=[("Martin",1,1) ; ("Dupond",2,1) ;("Dupuy",3,1) ;
("Durand",4,1) ;("Jean-Claude",5,2) ;("Brice",6,3) ;("Mr Dusse",7,3) ;("Pignon",8,4)] ; ;
enterprise  : (string * int * int) list =
 ["Martin", 1, 1 ; "Dupond", 2, 1 ; "Dupuy", 3, 1 ; "Durand", 4, 1 ;
  "Jean-Claude", 5, 2 ; "Brice", 6, 3 ; "Mr Dusse", 7, 3 ; "Pignon", 8, 4]
```

1. Écrire une fonction `nom` qui à un entier n associe le nom de l'employé numéro n de l'entreprise.
2. Écrire une fonction `lechef` qui à un nom d'employé et à la liste précédente représentant l'entreprise, associe le nom de son chef.

```
#lechef("Brice",enterprise) ; ;
-  : string = "Dupuy"
#lechef("Durand",enterprise) ; ;
-  : string = "Martin"
#lechef("Napoleon",enterprise) ; ;
-  : string = "Cette personne n'est pas dans l'entreprise !"
#lechef("Martin",enterprise) ; ;
-  : string = "C'est MOI le Chef !"
```