Examen de
Programmation fonctionnelle

Durée : 2 h

Tous documents autorisés

1 Bases

1. Soit le type

```
#type 'a t = A | B of int * 'a t;;
```

Donner les règles de typage pour les contructeurs `A` et `B`.

2. Soient les expression suivantes :

```
#let m = fun n m f x -> n (m f) x
#and d = fun f x -> f (f x) in
#let q = m d d in
#let s = m q q;;
```

Donner le type de m (sans justification) et argumenter pour justifier le fait que le calcul de `s` boucle ou ne boucle pas.

2 Programmation

1. Écrire la fonction

```
#enleve;;
 - : 'a -> 'a list -> 'a list = <fun>
```

qui supprime un élément d'une liste (si l'élément apparaît plusieurs fois dans la liste, seule la première occurrence est supprimée).

2. Écrire la fonction

```
#couples;;
- : 'a list -> ('a * 'a) list = <fun>
```

qui calcule l'ensemble des couples ordonnés d'éléments d'une liste l, c-à-d la liste de couples (x, y) où x apparaît avant y dans la liste l.

```
#couples [1;3;2;0];;
- : (int * int) list = [1, 3; 1, 2; 1, 0; 3, 2; 3, 0; 2, 0]
```

3. Écrire un programme de résolution du problème « Le compte est bon » : soit L un ensemble d'entiers et n un entier ; il s'agit d'essayer d'obtenir n avec une suite d'opérations arithmétiques (addition, soustraction, multiplication, division exacte) appliquées aux éléments de L, ceci sans utiliser deux fois le même nombre.

Par exemple pour $L = \{25, 4, 1, 5, 10, 3\}$ et $n = 213$, une solution possible est[1] :

$$25 + 4 = 29;\ 29 - 5 = 24;\ 10 - 1 = 9;\ 9 * 24 = 216;\ 216 - 3 = 213$$

Indications :
- Le programme doit chercher de façon aveugle, en essayant toutes les combinaisons.
- Le programme est récursif : choix de deux nombres dans la liste L, suppression de ceux-ci de la liste, choix d'une opération, ajout du résultat à la liste et récursion sur la liste obtenue (utiliser `couples` et `enleve`).
- Il y a plusieurs appels récursifs en séquence : à chaque étape, il faut essayer tous les couples et toutes les opérations (on pourra éventuellement utiliser l'itérateur `List.iter`).
- Quand une solution est trouvée, le programme peut stopper la recherche par la levée d'une exception (à rattraper) après avoir affiché la solution.

[1] Cette solution a été produite par un programme Ocaml de 33 lignes et 897 caractères.