Travaux Pratiques ML No.6

Cours d'Informatique de Deuxième Année

—Deug SSM M4—

Aujourd'hui...

L'organisation de ce TP est la suivante:

— Le mécanisme d'exception.
— Encore des arbres...

Les arbres

Dans les arbres à branchements finis (`ABF`), chaque nœud a un nombre indéterminé de fils (à chaque nœud on associe la liste de ses fils). La structure de données des arbres `ABF` est la suivante:

```
# type 'a ABF = Nil | Node of 'a * 'a ABF list ;;
```

▸ **Exercice 2**. Ecrire la fonction qui calcule la hauteur d'un `ABF`.

. . . . . .

. . . . . .

▸ **Exercice 3**. Ecrire la fonction qui calcule le nombre de nœuds d'un `ABF`.

. . . . . .

. . . . . .

▸ **Exercice 4**. Ecrire une fonction de type `'a list -> 'a ABF -> bool` qui retourne `true` si et seulement si la liste est un chemin de l'arbre.

. . . . . .

. . . . . .

▸ **Exercice 5**. Ecrire une fonction de type `'a ABF -> 'a list list` qui retourne la liste de tous les chemins de l'arbre.

. . . . . .

. . . . . .

On veut représenter les expressions arithmétiques par le type suivant:

```
# type Formule =
#           Plus of Formule * Formule
#         | Opp of Formule                    (* oppose d'une formule *)
#         | Mult of Formule * Formule
#         | Inv of Formule                    (* inverse d'une formule *)
#         | Exp of Formule | Ln of Formule
#         | Cos of Formule | Sin of Formule
#         | X                                 (* la variable *)
#         | Const of int ;;
```

▸ **Exercice 6**. Ecrire une fonction de dérivation pour une expression de type Formule.

. . . . . .

. . . . . .

. . . . . .

▸ **Exercice 7**. Ecrire un `pretty printer` d'expressions arithmétiques[(1)].

. . . . . .

. . . . . .

Les exceptions

Le type exception `exn` comprend entre autres le constructeur `Failure` d'arité `1` dont l'argument est de type `string`. Une expression du type

```
# try e with
#     exn1 -> r1
#   | exn2 -> r2
#   | ...
#   | exnn -> rn ;;
```

permet, si l'évaluation de `e` produit une exception $\texttt{exn}_i$, de retourner la valeur $\texttt{r}_i$; si l'évaluation de `e` n'a pas produit d'exception alors l'évaluation de `e` est renvoyée normalement.

▸ **Exercice 8**. Réécrire la fonction de division de deux entiers à partir de la fonction `/` de telle sorte qu'elle renvoie une valeur infinie `infinity` si son deuxième argument vaut `0`.

. . . . . .

▸ **Exercice 9**. Ecrire une fonction `find_index P l` qui renvoie la position du premier élément de la liste `l` qui vérifie le prédicat `P`. On utilisera des exceptions pour signifier que cette valeur existe ou n'existe pas.

. . . . . .

. . . . . .

Exercice supplémentaire

▸ **Exercice 10**. Les expressions en sortie du dérivateur sont simplifiables. Ecrire la fonction de simplification.

(1) on s'efforcera de ne pas abuser du parenthésage...