

# Série d'exercices #3 $\frac{1}{2}$

IFT-2035

September 24, 2009

## Mini évaluateur

Soit le code Haskell ci-dessous:

```
data Exp = Enum Int           -- Une constante
         | Eplus Exp Exp      -- e1 + e2
         | Etimes Exp Exp     -- e1 * e2
         | Eneg Exp           -- (- e)
         | Egt Exp Exp        -- e1 > e2
         | Enot Exp           -- (not e)
         | Eif Exp Exp Exp    -- if e1 then e2 else e3

data Val = Vnum Int
         | Vbool Bool

eval :: Exp -> Val
```

*Exp* est un type qui représente des expressions simples incluant des opérateurs arithmétiques et booléens. *Val* est un type qui représente des valeurs entières ou booléennes. *eval* est une fonction qui doit évaluer une expression et retourner la valeur qui en résulte.

1. Écrire la fonction *eval*
2. Remplacer dans *Exp* les expressions *Eplus*, *Etimes*, *Eneg*, *Egt*, *Enot* par une expressions générique d'appel de fonction de la forme *Ecall Exp [Exp]* où *Exp* sera la fonction à appeler et *[Exp]* la liste des arguments. Faites tous les changements nécessaires (y compris à *Exp* et *Val*) pour que votre évaluateur fonctionne comme avant, en utilisant des remplacements de la forme:

```
eplus e1 e2 = Ecall ?? [e1, e2]
etimes e1 e2 = Ecall ?? [e1, e2]
eneg e      = Ecall ?? [e]
egt e1 e2  = Ecall ?? [e1, e2]
enot e     = Ecall ?? [e]
```