Lambda calcul et programmation fonctionnelle

TP 1

Exercice 1

Nous allons utiliser le compilateur GHC (Glasgow Haskell Compiler) dans le cadre de ces TPs.

Le premier programme que nous allons écrire est le classique « Hello, world ».

Haskell étant un langage purement fonctionnel, afficher quelque chose à l'écran n'est pas aussi simple qu'avec les autres langages.

Ouvrez un fichier hello.hs avec votre éditeur de texte favori et entrez la fonction suivante :

```
main = putStrLn "Hello, world"
```

Puis compilez le programme avec la commande :

ghc hello.hs

Cette commande crée deux fichiers intermédiaires (hello.hi et hello.o), ainsi qu'un exécutable, hello. Lancez-le en exécutant la commande :

./hello

Quel est le type de la fonction main ? Nous ne l'avons pas donné, mais Haskell l'a inféré lui-même. Nous pouvons également l'ajouter en tête du fichier :

```
main :: IO ()
main = putStrLn "Hello, world"
```

Quel est ce type étrange ? On ne va pas en parler en détail, mais il dit « attention, cette fonction fait des entrées-sorties (IO), elle interagit avec le monde réel, elle n'est pas purement fonctionnelle ». C'est une manière d'encapsuler le monde réel dans le paradigme purement fonctionnel.

Que fait la fonction putStrLn ? Elle prend une chaîne de caractères en paramètre et l'affiche à l'écran. Elle ne peut être appelée que dans des fonctions qui manipulent le monde réel, comme la fonction main par exemple. Vous ne pouvez pas l'utiliser n'importe où !

Exercice 2

On va écrire un programme qui affiche l'inverse du nombre 10.

Créez un nouveau fichier, inverse.hs.

Écrivez la fonction inverse qui, à un nombre entier, associe un nombre réel, son inverse. N'oubliez pas d'écrire aussi la signature de la fonction, même si Haskell est capable de l'inférer.

Puis, écrivez la fonction main suivante :

```
main :: IO ()
main = putStrLn (inverse 10)
```

Compilez le programme. Que se passe-t-il ? Normalement, le compilateur doit refuser votre programme en raison d'une erreur de type. En fait, ici on ne peut pas utiliser putStrLn étant donné que l'argument n'est pas de type chaîne de caractère.

Nous allons donc plutôt utiliser la fonction print, qui est une fonction qui prend une valeur quelconque en paramètre et l'affiche à l'écran. Comme putStrLn, elle ne peut s'utiliser que dans une fonction qui manipule le monde réel, telle que main.

```
main :: IO ()
main = print (inverse 10)
```

Compilez et exécutez votre programme. Le résultat est-il correct ?

Modifiez ensuite votre programme pour qu'il calcule l'inverse de 0. Que se passe-t-il à la compilation ? À l'exécution ?

Modifiez ensuite votre programme pour qu'il calcule l'inverse de la chaîne de caractère « toto ». Que se passe-t-il à la compilation ?

Exercice 3

Dans cet exercice nous allons voir que les calculs sur des « réels » (en réalité les nombre en virgule flottante, c'est-à-dire ceux de type Float ou Double) peuvent tendre des pièges redoutables.

Écrivez un programme flottants.hs suivant :

```
main :: IO ()
main = print (0.0017 * 1000)
```

Qu'obtenez-vous ?

Maintenant, lancez le programme suivant (qui devrait donner le même résultat) :

```
main :: IO ()
main = print (0.0017 * 100)
```

Qu'obtenez-vous ? Les nombres flottants sont des estimations de nombres réels, parfois un calcul ne donne pas le résultat attendu mais une approximation. Cela peut donner des mauvaises surprises.

Lancez le programme suivant :

```
main :: IO ()
main = print ((0.0017 * 1000) == (0.0017 * 100 * 10))
```

On devrait afficher True. Qu'obtenez-vous ?

Exercice 4

Nous allons écrire un programme qui va calculer le trentième terme de la suite de Fibonacci.

La suite de Fibonacci est une suite entière. Elle est définie comme suit :

Fibo(0) = 0.
Fibo(1) = 1.
Fibo(n) = Fibo(n-1) + Fibo(n+1).

Créez un nouveau fichier fibo.hs. Écrivez une fonction qui permet de calculer la valeur de Fibo(n) pour une valeur de n donnée en paramètre.

Puis écrivez la fonction main qui permet de l'afficher :

```
main :: IO ()
main = print (fibo 30)
```

Compilez et exécutez votre programme. Vous devez obtenir la valeur 832040.

Calculez ensuite la valeur de Fibo(40). Que constatez-vous ?

Calculez la valeur de Fibo(-1). Que constatez-vous ?