Cours de programmation fonctionnelle

1. Paradigme

Pas seulement le langage Haskell.

1. Fonctions = Citoyens de première classe.

Les fonctions vont être des choses comme les autres dans le langage

(Ex : Variable qui est une fonction, fonction qui prenne des fonctions)

Fonction et entier sont sur un pied d’égalité.

1. Impératif = calcul est une suite d’états.

En fonctionnel le calcul est décomposé sous forme de valeur

1. Pureté = Pas d’effet de bord.

Langage fonctionnel :

* LISP (1950)
* OCAML, ML (1970)
* Scala, CLOJURE (S’exécute dans une JVM)
* F#
* ERLANG
* JAVA, C#

Haskell exemple :

Somme n = Sum[1…n]

Interprète HASKELL : ghci

: Constructeur de liste

x :xs liste de tête x queue xs -> s du pluriel liste

exemple :

module Mystere where

myst [] = []

myst (x :xs) = myst ys ++ [x] ++ myst zs

where ys = [ y | y <- xs, y < x ]

zs = [ z | z <- xs, z > x ]

prop\_myst xs = sort xs == myst xs