Exercices
Introduction à la programmation fonctionnelle

Exercice 1
Donnez les résultats des expressions ci-dessous.
(1) 'Z' < 'a'
(2) "abc" <= "ab"
(3) "abc" >= "ac"
(4) 1+2*3
(5) 5.0 - 4.2 / 2.1
(6) 3 > 4 || 5 < 6 && not (7 /= 8) (dites quelles expressions sont évaluées)
(7) if 6 < 10 then 6.0 else 10.0
(8) 0XaB + 2
(9) 0xBa + 2

Exercice 2
Toutes les expressions ci-dessous comportent des erreurs. Expliquez chacun.
(1) 18`mod`7/2
(2) if 2<3 then 3
(3) 1<2 and 5>3
(4) 6+7 div 2
(5) 4. + 3.5
(6) 1.0<2.0 or 3>4
(7) 1.0 = 3
(8) if 4 > 4.5 then 3.0 else 'a'

Exercice 3
Écrivez les expressions ci-dessous en Haskell sans utiliser les opérateurs logiques :
(1) E1 || E2
(2) E1 && E2

Exercice 4
Donnez les résultats des expressions ci-dessous.
(1) [1,2,3] !! ([1,2,3] !! 1)
(2) head[1,2,3]
(3) tail[1,2,3]
(4) "a":["b","c"]
(5) "abc" ++ "d"
(6) tail "abc" ++ "d"
(7) head "abc" : "d"
(8) ([1,2,3] !! 2 : []) ++ [3,4]
(9) [3,2]++[1,2,3]!!head[1,2]:[]

Exercice 5
Pourquoi les expressions suivantes contiennent-elles des erreurs?
(1) head[]
(2) tail[]
(3) ["n"]:["o","n","!"]
(4) 1 ++ 2
(5) head "abc" ++ "d"

Exercice 6
Donnez le type des expressions ci-dessous.
(1) (1.5,("3",[4,5]))
(2) [[1,2],[]]
(3) (['a','b'],[[],[1,2,3]])

Exercice 7
Donnez une expression qui satisfasse les types ci-dessous.
(1) [[[Integer]]]
(2) [(Integer, Char)]
(3) (Double, [[Integer]])
(4) (((Integer,Integer),[Bool],Double),(Double,Char))

Exercice 8
Écrire une fonction qui retourne le 2e élément d'une liste. La liste contient toujours au moins 2 éléments.

Exercice 9
Écrire une fonction qui retourne le 2e caractère d'une chaîne de caractère. La chaîne contient toujours au moins 2 caractères.

Exercice 10
Écrire une fonction qui réalise une rotation cyclique vers la gauche d'une liste : si l est [] le résultat est [];
si l contient un seul élément, le résultat est la liste l;
si l == $[a_1,a_2,...a_n]$, l'application sur l donne $[a_2,a_3,...a_n,a_1]$.

Exercice 11
Sans utiliser le sélecteur !!, écrire une fonction qui prend une liste et qui retourne la liste sans son second élément. La liste contient toujours au moins 2 éléments.

Exercice 12
Écrire une fonction qui réalise i rotations cycliques vers la gauche d'une liste, c.-à-d., si liste = $[a_1,a_2,...a_n]$, l'application de cette fonction sur liste donne $[a_{i+1},a_{i+2},...a_n,a_1,a_2,...a_i]$.

Exercice 13
Écrire une fonction qui duplique tous les éléments d'une liste. Pour liste = $[a_1,a_2,...a_n]$, l'application de cette fonction sur liste donne $[a_1,a_1,a_2,a_2,...a_n,a_n]$.

Exercice 14
Comment Haskell déduit-il les types des parametres de la fonction ci-dessous ?

```
exercice3 a b c d =
        if a == b then c+1 else if a > b then c else b+d
```

Exercice 15
Écrire une fonction qui réalise un tri fusion de 2 listes triées. Par exemple, pour liste1 = [1,2,4] et liste2 = [2,5], la fonction retourne [1,2,2,4,5].

Exercice 16
Écrire une fonction qui détermine si une liste passée en paramètre est vide.

Exercice 17
Écrire une fonction qui réalise une rotation cyclique vers la gauche d'une liste :
- si liste est [] le résultat est [];
- si liste contient un seul élément, le résultat est cet élément;
- si liste = $[a_1,a_2,...a_n]$, l'application sur liste donne $[a_2,a_3,...a_n,a_1]$.

Exercice 18
Écrire une fonction qui duplique tous les éléments d'une liste.
Pour liste = $[a_1,a_2,...a_n]$, l'application de cette fonction sur liste donne $[a_1,a_1,a_2,a_2,...a_n,a_n]$.

Exercice 19
Écrire une fonction qui résout le calcul récursif d'un entier positif élevé au carré sans utiliser l'opérateur de multiplication.
Condition d'arrêt : $0^2 = 0$
Pas inductif : $n^2 = (n-1)^2 + 2\ n-1$

Exercice 20
Écrire une fonction prenant un élément a et une liste liste comportant des listes de même type que a. La fonction retourne une nouvelle liste et qui insert a devant chaque liste contenue dans liste.
Par exemple, si liste = [[2,3],[],[4]] et a = 1, la liste retournée est [[1,2,3],[1],[1,4]]

Exercice 21
Écrire une fonction qui prend une liste d'éléments ordonnés et qui détermine l'élément le plus grand. Par hypothèse, la liste contient au moins un élément.

Exercice 22
Écrire une fonction qui retourne un tuple de 2 éléments conte- nant la somme des éléments indicés impairs et pairs res- pectivement. Par exemple, pour $[a_1,a_2, a_3, a_4, ...]$, le tuple $(a_1+a_3+..., a_2+a_4+...)$ est retourné. Par hypothèse, la liste contient au moins 2 éléments.

Exercice 23
Écrire une fonction qui calcule, pour un réel $x > 0$ et un entier $i > 0$, la valeur de x^{2i}. Remarquez que, pour $i = 3$, le calcul est équivalent à $((x^2)^2)^2$.

Exercice 24
Écrire une fonction qui produit la concaténation de 2 listes sans utiliser l'opérateur ++ qui est dédié à cet effet.
Votre fonction devra prendre un temps proportionnel à la lon- gueur de l'une des listes passées en paramètre.

Exercice 25
Écrire une fonction qui réalise i rotations cycliques vers la gauche d'une liste, c.-à-d., si liste = $[a_1,a_2, ... a_n]$, l'application de cette fonction sur liste donne $[a_{i+1},a_{i+2}, ... a_n, a_1, a_2, ... a_i]$.
On supposera que la liste n'est pas vide et que i est inférieur à la longueur de la liste.
Votre fonction devra prendre un temps proportionnel à la longueur de la liste.

Exercice 26
Réalisez une fonction qui incrémente tous les éléments d'une liste d'entiers.

Exercice 27
Réalisez une fonction qui remplace tous les nombres < 0 par 0 dans une liste. Les éléments > 0 demeurent inchangés.

Exercice 28
Réalisez une fonction qui retourne la plus grande valeur d'une liste d'entiers. La valeur 0 est retournée si la liste est vide.

Exercice 29
Réalisez une fonction qui prend une liste formée de chaînes de caractères et qui retourne la même liste mais ayant ses éléments ne dépassant pas 5 caractères de long. Les chaînes plus longues que 5 sont tronquées.

Exercice 30
La performance de la fonction integraleTrapeze peut s'améliorer si l'intervalle delta se calcule une seule fois et si le nombre d'évaluation de la fonction f est minimisé. Réécrire integraleTrapeze en tenant compte de cette suggestion.

Exercice 31
Quels sont les ensembles ZF définis par les expressions suivantes ?

(a) [(x,y) | x <- [1..2], y <- [2,5], x + y /= 4]
(b)[(x,y) | x <- [1..4], mod x 2 == 0, y <- "ab"]

Exercice 32
Définissez l'ensemble ZF donnant les listes ci-dessous.
(a) [1,2,3,4,5,6,7,8,10,11,12,13]
(b) [2,-3,4,-5,6,-7,8,-9,10,-11,12,-13]

Exercice 33
En utilisant les ensembles ZF, réalisez une implémentation de l'algorithme du tri rapide qui tri une liste.

Exercice 34
Écrire une fonction currifiée, appliqueListe, prenant une liste de fonctions et une valeur et qui applique chaque fonction à la valeur passée comme paramètre.

Exercice 35
1. Écrire une fonction qui détermine si une sous-chaîne est contenue dans une chaîne de caractères. Il est préférable de currifier cette fonction.
En utilisant projeteFonction, réaliser une liste de fonc- tions à partir d'une liste de chaînes de caractères [s1,s2,...sn]
de telle sorte qu'une fonction fi de la liste vérifie si la sous- chaîne si est contenue dans une chaîne de caractères.
2. Instancier cette fonction avec la liste ["a","bout","a bout"].
3. Que donne appliqueListe avec la liste obtenue et la chaîne "aboutir"?

Exercice 36
Écrire une fonction curryfie qui convertit une fonction prenant un tuple de 2 champs dans sa forme curryfiée.

Exercice 37
Écrire une fonction decurryfie qui convertit une fonction définie sous une forme curryfiée dans une forme prenant un tuple. Le nombre de paramètres de la fonction à décurryfier est 2.

Exercice 38
Réalisez une fonction qui retourne la suite infinie des nombres
(a) factoriels [0!,1!,2!,..]
(b) de Fibonacci [0,1,1,2,3,5,8,..]

Exercice 39
Réalisez une fonction qui retourne la somme des termes
[a0, a0 + a1, a0 + a1 + a2,..] à partir de la liste infinie [a0, a1, a2,..].

Exercice 40

Déterminer la racine carrée d'un nombre x par une approximation successive de nombres provenant de l'équation de récurrence de Newton-Raphson :

$a_0 = x / 2$

$a_n = (a_{n-1} + x/a_{n-1})/2$

(a) Réalisez la fonction newtonRaphson qui retourne la suite infinie des valeurs $[a_0, a_1, a_2, ..]$:

```
newtonRaphson :: double -> [double]
```

(b) Réalisez la fonction precision qui retourne la première valeur an satisfaisant $|an - an\text{-}1| < \varepsilon$ quand les ai proviennent d'une liste et que ε est un paramètre :

```
precision :: double -> [double] -> double
```

Pour trouver la racine carrée qu'un nombre, on devrait pouvoir faire

```
precision epsilon (newtonRaphson x)
```