CORRECTION SESSION NORMALE
PROGRAMMATION FONCTIONNELLE INF 112

Proposez Par : GROUPE GENIUS R

Par : Joël_Yk & Petnga Njemfa Steploic

Exercice 1:

1=>Reponses:

- La différence entre **Evaluation paresseuse** et **Evaluation Stricte** est que : **l'évaluation paresseuse** est une technique d'implémentation des programmes récursifs pour laquelle l'évaluation d'un paramètre de fonction ne se fait pas avant que les résultats de cette évaluation ne soient réellement nécessaires tandis que l'évaluation stricte est un mode d'évaluation où l'expression est évalué dès qu'elle peut-être liée à une variable.

-La différence entre **Typage Statique** et **Typage Dynamique** est que : **le typage statique** est une technique utilisée dans les langages fonctionnels pour associer à une fonction le type de son paramètre et le type de la valeur calculée tandis que **le typage Dynamique** consiste à laisser l'ordinateur réaliser l'ordinateur réaliser l'opération de Typage lors de l'exécution du code.

- La différence entre **Langage Fonctionnel** et **Langage Procédural** est que : **les langages procéduraux** sont des langages dans lesquels sont explicitement énoncées l'ensemble des instructions à accomplir pour réaliser une tache donnée tandis que celui **fonctionnel** est affranchit des effets de bord en interdisant toute opération d'assignation.

2=> **Une expression** est en **forme canonique** lorsqu'elle est sous forme la plus simplifiée possible.

3=> {

Réduction totale :
e = square (3 + 4)
= (3 + 4) * (3 + 4)

```
    = 7 * 7
    = 49
}
```

Exercice 2 :

1.) Donnons le type de chaque fonction :

```
twice :: (t -> t) -> t -> t
squares :: [Integer] -> [Integer]
sommeProduit :: (Foldable t1, Num t) => t1 t -> (t, t)
```

2.) Donnons la valeur des expressions

exp1 = twice (/2) 56 = 14

exp2 = map even (squares [2,3,5,6,8,10]) =
[True,False,False,True,True,True]

exp3 = snd (sommeProduit [2,3,5,6,8,10]) = 28800

3.) Ecrivons cette expression

```
let triangles = [ (a,b,c) | c <- [1..10], b <- [1..10], a <- [1..10] ]let
rightTriangles' = [ (a,b,c) | c <- [1..10], b <- [1..c], a <- [1..b], a^2
+ b^2 == c^2, a+b+c == 24]
```

Problème :

1.) Equivalent de la liste

```
Cons {head_ = 1, tail_ = Cons {head_ = 8, tail_ = Cons {head_ = 9,
tail_ = Cons {head_ = 3, tail_ = Nil } } } }
```

2.) Type des Expressions

```
Nil :: List t
Cons :: t -> List t -> List t
head_ :: List t -> t
tail_ :: List t -> List t
```

3.) Proposons les fonctions

```
fromList::List a->[a]
fromList Nil = [ ]
fromList (Cons x xs) = x:(fromList xs)

toList::[a]->List a
toList [ ] = Nil
toList (x:xs) = Cons x (toList xs)
```

4.)Type des fonctions

```
applyMap :: Fractional b => [List b] -> [List b]
aList :: List (List Integer)
```

5.)Donnons une fonction équivalente :

```
applyMap_ :: Fractional b => [List b] -> [List b]
applyMap_ [ ] = [ ] applyMap_ (x:xs) = [map_ (/2) x] ++
applyMap_ xs
```