

TD numéro 8

Des fonctions en argument et en résultat

Exercice 1

- Écrire une fonction qui étant donné une liste `l` (les éléments sont des atomes) et un prédicat unaire `p` (fonction à un argument rendant un résultat booléen) renvoie la liste des éléments de `l` qui vérifient `p`.

```
(verifie '(1 2 7 5 4) even?) → (2 4)
```

Exercice 2

On considère deux fonctions composables unaires `f` et `g` de `E` dans `E`.

- Écrire la fonction `comp` qui renvoie le résultat de la composition de ces deux fonctions appliquée à un élément. Par exemple si `f` et `g` sont deux fonctions de `E` dans `E` et `x` un élément de `E`, la fonction `comp` renverra l'élément `f(g(x))`.

```
(comp carre cube 2) → 64
```

- Écrire la fonction `fn` qui, étant donné une fonction `f` de `E` dans `E`, un entier `n` et `x` un élément de `E`, renvoie l'élément `fn(x)`, où `fn` est la composée `n` fois de `f`.

```
(fn carre 3 2) → 256
```

- Que fait la fonction `compf` ainsi définie :

```
(define compf
  (lambda (f g)
    (lambda (x)
      (f (g x)))))
```

- Écrire une version de `compf` utilisant `comp`.
- Écrire une fonction `fnf` qui, étant donné une fonction `f` et un entier `n`, renvoie `fn`.

```
(define puissance4 (fnf carre 2))
```

Exercice 3

- En utilisant la fonction abstraite sur les arbres vue en cours, écrire une fonction qui compte le nombre de valeurs paires dans un arbre de nombres.
- En adaptant la spécification de la fonction abstraite sur les arbres vue en cours, écrire une fonction qui multiplie par deux toutes les valeurs d'un arbre de nombres.

Exercice 4

- Définir la spécification de la fonction mystère ci-dessous :

```
(define mystere
  (lambda (a b)
    (if (null? b)
        '()
        (cons (a (car b)) (mystere a (cdr b))))))
```