IA01 — TD01

---

1 Évaluation d'expressions

Évaluer les expressions suivantes et comprendre le résultat obtenu.

```
23
(quote 23)
'23
(set x 32)
(setq x 32)
(list x x 32)
(cadr (list x x 32))
(setq x 'y)
(setq xx 5)
(setq y '(+ xx xx 32))
x
(eval x)
(eval y)
(cadr y)
(eval (list '+ (eval y)(cadr y)))
(setq z (+ (if x 2 0) (caddr y) (* y y)))
(setq y (list (setq z "Albert") x y))
z
y
(setq x (* x x))
x
```

2 Représentation d'un arbre binaire

Soit l'arbre suivant :

```
   a
 /   \
b     c
    /   \
   d     e
```

Proposer une représentation sous forme de liste.

Dessiner l'arbre binaire correspondant à l'expression (infixée) : `((x + 2) / (x - 3))`

Récrire cette expression en préfixé.

Ecrire une fonction LISP permettant de faire une transformation infixe → préfixe sur l'expression précédente, puis sur tout type d'expression correctement écrite.