TP MELOG n°III: application des structures arborescentes aux traitements des expressions arithmétiques et fonctionnelles

---

L'objectif de ce TP est d'appliquer les principes de la structuration des données pour mettre en œuvre un programme de traitement des expressions arithmétiques et fonctionnelles.

Les expressions arithmétiques se représentent simplement par des arbres comme (figure 1) l'expression `(5+(2+3))*((10*10)+(9*9))` qui s'écrit `(*(+ 5(* 2 3))(+(* 10 10)(* 9 9)))` en forme préfixée. les feuilles contiennent des valeurs numériques et les nœuds (binaires) représentent le symbole d'opérateurs.

Les opérations prises en compte sont la somme, la différence, le produit, le quotient et la puissance.

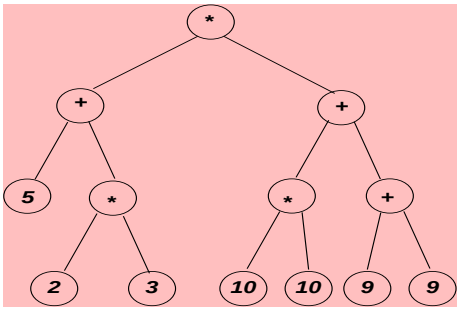

Figure 1 : Représentation d'une expression arithmétique par un arbre

Des tests pourront être effectués grâce à l'utilisation de classes mises à disposition pour la transformation d'expressions de leur forme textuelle à leur format d'arbre selon la classe `Arbre` elle aussi fournie.

1 Expressions arithmétiques

Question :

- À partir d'un arbre représentant une expression arithmétique, concevoir l'algorithme et écrire le programme qui permet d'évaluer cette expression.

2 Expressions polynomiales

Les nœuds de l'arbre peuvent aussi bien contenir le symbole d'une variable dans l'expression d'une fonction. On se limitera, dans un premier temps, à des fonctions n'ayant qu'une seule variable et que des monômes.

Questions :

Concevoir les algorithmes et écrire les programmes qui permettent :

- d'obtenir l'arbre qui représente la dérivée d'une expression polynomiale,
- d'évaluer l'expression pour une valeur de la variable.

3 Question bonus

Maintenant, les nœuds peuvent aussi contenir en plus le symbole d'une fonction trigonométrique de la variable considérée. On connaît les dérivées de ces fonctions trigonométriques[1].

Question :

- Concevoir l'algorithme et écrire le programme qui permet d'obtenir l'arbre représentant la dérivée d'une fonction représentée par un arbre donné et ainsi constitué.

---

[1] On pourra se limiter à des fonctions du type `sin(x)` (sans tenir compte des fonctions telles que `sin(`a`.x`^{n}`)`).