# Laboratoire : Les arbres AVL

## 1 Objectifs

Le but de ce laboratoire est d'améliorer les arbres binaires en imposant la restriction d'être un arbre AVL.

## 2 Travail à faire

Vous devez implémenter la classe que nous vous fournissons avec cet énoncé. La classe `Arbre` de ce laboratoire est une amélioration de la classe `Arbre` du laboratoire précédent. En particulier, il y a trois principales fonctions qui font en sorte que la classe respecte la condition d'arbre AVL :

```
// Indique si l'arbre est AVL.
bool estAVL() const;

// Ajoute un noeud en conservant la condition AVL.
void insererAVL(const E &);

// Enlève un noeud en conservant la condition AVL.
void enleverAVL(const E &);
```

Veuillez noter que les autres méthodes que nous vous demandons d'implémenter sont pratiquement identiques aux méthodes de la classe `Arbre` de la semaine dernière. C'est pour vous une occasion de réutiliser votre code pour vous épargner du travail.

Pour ce laboratoire, il est particulièrement important d'utiliser les méthodes utilitaires privées. Par exemple, dans la solution, la fonction `zigzag` ne fait que 2 lignes de code. Cela facilitera le débogage de votre programme et sa lisibilité.

# 3 Documentation

Voir la section Documentation/Normes sur le site Web du cours. Vous y trouverez la description des commentaires attendus dans un programme ainsi que des normes de programmation en vigueur dans notre cours.

# 4 Important

1. Nous vous fournissons des tests unitaires *Google Test* que vous pouvez utiliser pour vérifier votre implémentation.
2. Vous êtes tenu de faire la gestion des exceptions dans les méthodes que vous avez à implémenter. Référez-vous à la documentation *Doxygen* fournie avec l'énoncé pour connaître les types d'exception que vous avez à gérer pour chacune d'elles. Vous devez utiliser le cadre de la théorie du contrat que nous avons préparé dans les fichiers `ContratException.cpp` et `ContratException.h` disponibles sur le site du cours.
3. Vous devez documenter chaque méthode, que vous avez à implémenter, avec les commentaires de *Doxygen*. Référez-vous à section de Doxygen sur le site Web du cours ainsi qu'aux exemples de cette semaine pour découvrir les commentaires que vous avez à écrire. Assurez-vous de respecter les normes de programmation en vigueur disponibles également sur le site du cours.
4. Vous devez générer la documentation en format HTML. À cette fin, nous vous fournissons un fichier de configuration `sdd.doxyfile` que vous pouvez utiliser tel quel.
5. Nous vous encourageons à utiliser au maximum la partie privée de la classe afin d'y ajouter des fonctions utilitaires privées. Elles permettent d'augmenter la lisibilité du code et de réduire la duplication de code identique.

# Bon travail !