

## TD 4 : Arbres AVL

### 1 AVL : manipulation des algorithmes du cours

QUESTION 1 – Construire l'AVL obtenu par ajout successif à l'arbre vide des valeurs suivantes : 25, 60, 35, 10, 5, 20, 65, 45, 70, 40, 50, 55, 30, 15. On ne représentera que les étapes nécessitant une rotation pour rétablir l'invariant d'équilibrage.

QUESTION 2 – Retirer à l'arbre obtenu 25, 30 puis finalement 35.

### 2 Arbres AVL : stockage de la hauteur

En cours on a vu que la hauteur  $h$  d'un arbre AVL est dominée par  $\log n$ , où  $n$  est la taille de l'arbre. L'utilisation de cette classe d'arbres promet un gain en efficacité pour les fonctions AJOUTER et SUPPRIMER, qui ne doit pas être perdu par des calculs répétés de la hauteur des sous-arbres.

QUESTION 3 – Proposer une implémentation de ces fonctions (et des fonctions auxiliaires de rotation et de ré-équilibrage) qui, pourvu que la hauteur de chaque nœud  $x$  de l'arbre soit initialement correcte, maintient le champ  $x.hauteur$  à jour.

### 3 Concaténation de deux arbres AVL

On considère ici deux arbres AVL  $T_1$  et  $T_2$  tels que tous les éléments de  $T_1$  sont plus petits que les éléments de  $T_2$ .

QUESTION 4 – Notons  $H$  la hauteur du plus grand des deux arbres. Sans écrire de pseudo-code, expliquer comment on peut construire un arbre AVL contenant exactement les éléments de  $T_1$  et de  $T_2$  (un tel arbre s'appelle la concaténation de  $T_1$  et  $T_2$ , que l'on peut noter  $T_1 \cup T_2$ ) en  $O(H)$ .

*Indication* : on pourra utiliser les procédures suivantes vues en cours :

- AJOUTER( $e, T$ )
- SUPPRIMER( $e, T$ )
- RÉ-EQUILIBRER( $T$ )

### 4 Section de deux arbres AVL

QUESTION 5 – Étant donné un arbre AVL  $T$  et une valeur  $x$ , écrire une fonction SECTION qui renvoie une paire d'AVL  $T_1$  et  $T_2$  comprenant les valeurs contenues dans  $T$  respectivement strictement plus petites et strictement plus grandes que  $x$ .