TP 5 – Équilibrage et arbres AVL

On se propose dans ce TP de mettre en œuvre un arbre AVL. Pour cela, on se basera sur une implémentation précédente d'un arbre binaire de recherche d'entiers (cf. TP3).

Travail à faire

Compléter le menu fourni en écrivant les fonctions qui effectuent les traitements suivants :

1. Effectuer une rotation gauche de l'arbre à partir d'un nœud saisi par l'utilisateur. L'arbre sera affiché avant et après rotation. (On ne demande pas à ce que l'arbre soit équilibré dans ce cas).
2. Idem pour une rotation droite.
3. Créer une fonction d'ajout qui s'assure que l'arbre est équilibré
4. Ajouter une fonction equilibrer qui, à partir d'un arbre quelconque (non AVL), le transforme en arbre équilibré. Cette fonction n'utilisera que des rotations pour parvenir à ses fins. L'algorithme à exploiter est le suivant :

```
Si arbre déséquilibré à droite {
  Si le SAD est déséquilibré à gauche {
    Effectuer une double rotation gauche-droite
  } Sinon { Effectuer simple rotation gauche }
}
Sinon si arbre déséquilibré à gauche {
  Si le SAG est déséquilibré à droite {
    Effectuer une double rotation droite-gauche
  } Sinon { Effectuer une simple rotation droite }
}
```

Prévoir des messages indiquant à l'utilisateur le résultat des traitements effectués et les erreurs de saisie (lorsqu'une valeur est absente de l'arbre).

Fichiers fournis

- Le module **ab{.c,.h}** contient les primitives d'accès à un arbre binaire d'entiers.
- Le fichier **tp_AVL.c** contient le menu principal.

Exercices complémentaires

1. Écrire une fonction oter_noeud qui enlève un nœud de l'arbre saisi par l'utilisateur et effectue un équilibrage si nécessaire. Afficher le nombre de rotations effectuées dans ce cas.