Algorithmique et programmation procédurale

TP7 – Rotations et arbres binaires de recherche équilibrés (AVL)

Pour faciliter l'implantation des arbres AVL, vous pouvez utiliser la structure définie pour les arbres ABR en ajoutant le champ `hauteur` comme suite :

```
typedef struct AVL {
      int racine;
      struct AVL* filsG;
      struct AVL* filsD;
      int hauteur;
} AVL;
```

a) Définir une bibliothèque d'AVL (AVL.c et AVL.h) en implantant les opérations suivantes :

1. `AVL* rotationGauche(AVL* a)`: faire une rotation à gauche en mettant à jour la hauteur de nouvel arbre ;
2. `AVL* rotationDroite(AVL* a)`: faire une rotation à droite en mettant à jour la hauteur de nouvel arbre ;
3. `AVL* rotationGaucheDroite(AVL* a)`: faire une rotation gauche droite ;
4. `AVL* rotationDroiteGauche (AVL* a)`: faire une rotation droite gauche ;
5. `AVL* inserer(AVL* a, int x)`: ajouter en feuille un sommet contenant l'entier x puis renvoie le nouvel arbre ;
6. `AVL* supprimer(AVL* a, int x)`: supprimer le sommet qui contient x ;

Note : il faut réécrire vous-même d'autres fonctions de base (recherche, parcours, max, supprimerMax, ….) comme avec les arbres ABR qui permettent d'implanter toutes ces opérations.

b) Écrire un programme principal qui inclut votre bibliothèque et permet de tester ses fonctions. À partir d'un arbre AVL vide au démarrage, l'utilisateur doit être capable d'effectuer via un menu textuel les opérations de l'exercice précédent. Après chaque action, votre programme affiche l'état actuel de l'arbre.