TP numéro 2
Arbres binaires de recherches, Arbres AVL

Programmation impérative avancée, ENSIIE

Semestre 2, 2015–16

On reprend l'interface des dictionnaires de la séance précédente, ainsi que le fichier de tests. On complétera le `Makefile`.

Exercice 1 : Arbres binaires de recherche

1. Dans un fichier `abr.c`, implémenter les dictionnaires à l'aide d'arbres binaires de recherche.
2. Modifier le `Makefile` pour permettre la compilation du programme de test `test_abr` utilisant cette implémentation.
3. Compiler, tester.

Exercice 2 : Arbres AVL

4. Recopier `abr.c` dans un fichier `avl.c`, et modifier le pour implémenter les dictionnaires à l'aide d'arbres AVL.
5. Modifier le `Makefile` pour permettre la compilation du programme de test `test_avl` utilisant cette implémentation.
6. Compiler, tester.

Exercice 3 : Comparaison de complexité

7. Modifier le fichier `test.c` pour effectuer les opérations suivantes, où n est un entier passer en paramètre au programme :
   - Créer un dictionnaire de taille $\frac{n}{10}$.
   - Pour i allant successivement de 0 à $n-1$, insérer une association entre i et la chaîne contenant sa valeur.
   - Supprimer $\frac{n}{2}$ éléments associés à un entier choisi aléatoirement entre 0 et $n-1$. On libérera la mémoire allouée pour la chaîne de caractère.
   - Afficher, pour chaque entier entre 0 et $n-1$, soit la valeur associée dans le dictionnaire si elle existe, soit un message disant qu'elle n'existe pas.
8. Instrumenter le code de `test.c` avec la fonction `clock()` (cf. man) pour afficher le temps moyen en μs nécessaire pour réaliser chacune des opérations d'insertion, de recherche et de suppression. (On enlèvera les affichages.)
9. Compiler et comparer les trois implémentations des dictionnaires.