Assignment Two

Objectives

- Understand how the avl tree works
- Give you further practice with C and data structures

Admin

Marks 10 marks, excluding bonus marks. Marking is based on the correctness and efficiency of your code. Your code must be well commented.
Group? This assignment is completed individually.
Due Time ~~23:59:59pm on Saturday 28 April 2018.~~ 23:59:59pm on Wed 2 May 2018.
Late Submissions Late submissions will not be accepted!

In this assignment, you will implement avl tree and a set of functions associated with avl tree. For simplicity, we assume that each item contains an integer key and an integer value, and **all the keys of an avl tree are distinct**. A template file named MyAVLTree.c is provided. MyAVLTree.c contains the type definitions of avl tree and avl tree node as well as some basic functions.

You need to implement the following functions:

1. AVLTree *CreateAVLTree(const char *filename). This function creates an avl tree by reading all the items from a text file or from the standard input (keyboard) depending on the argument filename. If filename is "stdin", this function will read all the items from the standard input. Otherwise, it will read all the items from a text file with filename as its full path name.

   An input text file contains zero or more items where each item is of the form (key, value). Any characters such as white space between two adjacent items are ignored. For example, the following sample file contains 10 items:

   (2, 50) (4, 30) (9, 30) (10, 400) (-5, -40)
   (7, 20) (19, 200) (20, 50) (-18, -200) (-2, 29)

   Similarly, when reading from the standard input, each input line may have zero or more items, separated by one or more white space characters. An empty line indicates the end of input.

   In case of an error in the input, this function will print the error and your program terminates.

2. AVLTree *CloneAVLTree(AVLTree *T). This function creates an identical copy (clone) of the input avl tree T, and returns a pointer to the clone tree.

3. AVLTree *MergeTwoAVLTrees(AVLTree *T1, AVLTree *T2). This function merges two avl trees T1 and T2 into a new tree, destroys T1 and T2, and returns the new tree. When destroying T1 and T2, it frees all the heap space occupied by T1 and T2. Note that if you reuse one of T1 and T2, you cannot free its heap space.

   We assume that all the keys in T1 and T2 are distinct.

   Bonus marks: A correct merge function with the time complexity O(m+n) will be awarded **2 bonus marks**, where m and n are the sizes of T1 and T2, respectively.

4. int InsertNode(AVLTree *T, int k, int v). If the key k exists in the tree, this function simply returns 0 without adding the new item (k, v) to the tree. Otherwise, it inserts the new item (k, v) into the avl tree T, increases the tree size by one and returns 1.

5. int DeleteNode(AVLTree *T, int k). If the key k exists in the avl tree T, this function deletes the associated node, decreases the tree size by one and returns 1. Otherwise, it returns 0 only.

6. AVLTreeNode *Search(AVLTree *T, int k). This function search for the item with key k. If it finds k in the tree T, it returns a pointer to the node containing the key k. Otherwise, it returns NULL.

7. void FreeAVLTree(AVLTree *T). This function frees up the heap space occupied by the AVL tree T.

8. void PrintAVLTree(AVLTree *T). This function prints all the items stored in the avl tree T sorted in increasing order of keys on the standard output (screen). Each item is denoted by (key, value) with one item per line.

For each function, analyze its time complexity, and put the time complexity analysis as comments before the function.

How to submit your code?

a. Go to the assignment two page
b. Click on Make Submission
c. Submit your MyAVLTree.c file that contains all the code.

Plagiarism

This is an individual assignment. Each student will have to develop their own solution without help from other people. In particular, it is not permitted to exchange code or pseudocode. You are not allowed to use code developed by persons other than yourself. All work submitted for assessment must be entirely your own work. We regard unacknowledged copying of material, in whole or part, as an extremely serious offence. For further information, see the Course Information.