Lab 7: AVL Tree

7.1. For an AVL tree to store integer values, implement the following requirement:

- Create data structure for AVL Tree
- Initialize empty tree
- Insert some value to tree
- Print tree to console screen in pre-order, in-order, post-order
- Delete some value from the tree
- Calculate tree height.
- Identify whether a given value exists in the tree

Students must present some functions such as single rotate left, single rotate right, double rotate left, double rotate right to balance the tree.

7.2 Write a program to determine if given binary trees are AVL trees. Suppose that trees only store integer number types.

Your program read trees in input.txt and output yes/no AVL Tree. In input.txt, the first row is the number of the tree. The next rows are trees, with each tree will be on a line and in pre-order. Each integer value is separated by a space.

For example,
input.txt:
3
1 2 3 4 5
5 4 3 2 1
4 2 1 3 5

Ouput.txt:
No
No
Yes

7.3. Write a program to test whether all leaves of an AVL tree have the same depth

7.4. Write a program to find the least common ancestor for any two given nodes in AVL.

7.5

Refer back Lab 3 Sorting and search for detail description. In Lab 3, you do a simple spell checker with Linear Search and Binary search. In Lab 5, you continue that project by adding a new search – Search with Hash Table. Now in this Lab 7, you will do search with Binary Tree and AVL Tree

Compare the result from different methods. Analyze and discussion.

Methods	Run time (ms)
Linear Search	
Binary search	
Hashing 1	
Hashing 2,…	
Binary Tree	
AVL Tree	

What is average depth, longest depth of nodes in these Binary and AVL Tree?