# Theory Homework 2

Data Structures and Algorithms in JAVA
Shlomo Hershkop
Department of Computer Science
Columbia University
Fall 2013

**Out:** Oct 9 2013
**Due:** Sunday Oct 20, 11pm.

Goal: review stack, list, and trees

1. Suppose you have a linked list with header and tail. Is there any advantage for creating items with space for 8 items per link ?
   That is, instead of storing single elements with links, each node would have 4 spaces.
   for example:
   public class LinkedNode8 {
    private int data[8];
    private LinkedNode4 next;

   and you would have specific getters and setters to check all 8 locations etc

   how would runtimes for insert, find, and delete be affected for the linked list class ??

2. Show the resulting stack and the output of the following series of stack operations: push(15), push(28), pop(), push(3), push(81), pop(), pop(), push(9), push(1), pop(), push(7), push(4), pop(), pop(), push(4), pop(). Assume that each popped value is printed.

3. When we have a list of items, one way of saving time when deleting an item in a collection, is to mark the item as deleted. That is you need to add another field variable/node saying if the list item is active/alive. This is known as “Lazy Deletion”. In addition the list needs to keep track of how many elements are alive and how many are not. When the number of dead items reaches some point, you can run through the list and actually delete them..
    1. list the advantages and disadvantages of lazy deletion
        1. for array implementation of lists
        2. for linked list implementation

2. how would this trick affect binary trees operations.

4. Define "degree" of a node as the number of its non-empty children. Prove by induction that the number of degree 2 nodes in any binary tree is one less than the number of leaves.

5. A binary search tree is created by inserting, in order, the following numbers: 18, 3, 25, 0, 222, 2, 11, 12, 34, 30, 40. Draw the tree at each insert. Then give the pre-order, post-order and in-order traversals of the tree.

6. Give an O(n) algorithm for computing the depth of all the nodes of a tree T, where n is the number of nodes of T. Explain how your algorithm is O(n).

7. Describe in pseudo-code a non-recursive method for performing an in-order traversal of a binary tree in linear time. Make sure to show runtimes (big O)

8. Show by drawing the end result of adding 14, 4, 13, 2, 15, 18, 17, 39 into an initially empty binary search tree (one tree). Then show the result of deleting the root (another tree).

9. Show (Draw) the results of inserting 2, 1, 4, 5, 9, 10, 11 into an initially empty AVL tree. Now insert 13, 3, 17, 33 and show the new tree

10. Draw the B-tree with M=5, L=7 resulting from inserting the following keys (in this order) into an initially empty tree: 4, 40, 23, 50, 11, 34, 62, 78, 66, 22, 90, 59, 25, 72, 64, 77, 10, 12.

11. Two binary trees are similar if they are both empty or have similar left and right subtrees (not the values but the left/right children subtree shapes). Write some psuedo code isSimilar to take 2 binary trees and decide if they are similar. What is the running time for the isSimilar code ?

    public boolean isSimilar (BST a, BST b);

12. In your own words, compare a Splay tree to AVL tree.