# Data Structures Practice Problems for Placements

## General Algorithms (will be discussed on 18/08/12)

1. merge two sorted arrays in place
2. given two sorted arrays a1 and a2 of lengths l1 and l2, l1 < l2. last l1 slots in a2 are vacant. merge a1 and a2 into a2 without any extra space.
3. array of size n with numbers from 1 to n, one number repeated, one number not present. find the missing number.
4. given an array of numbers, find the number that occurs more than n/2 times in O(n) time and O(1) space
5. how to "shuffle" an array - Fischer-Yates Shuffle/Knuth Shuffle
6. given 2 sorted arrays, find the kth smallest number.
7. given balls of k colors in an array, sort them into clusters of different colors.
8. Given n red balls and m blue balls and some containers, how would you distribute those balls among the containers such that the probability of picking a red ball is maximized, assuming that the user randomly chooses a container and then randomly picks a ball from that.

## Arrays

1. given sorted arrays a1 and a2. find i,j s.t. a1[i]+a2[j] <= n and their sum is highest possible.
2. a sorted array is cycled by k. search for an element
3. given integers 1 to 100 in an array of 99, with one number missing. find the missing number. what if the given integers are from 60001 to 60100 in an array of size of 99.
4. how to reverse the words in a sentence without any extra space.

## Binary Search Tree

1. find the diameter of a tree (the longest path between any 2 nodes)
    - diameter of tree = Max (diameter of left subtree, diameter of right subtree, height(left-subtree) + 1 + height(right-subtree) + 1)
    - Diameter of a leaf node = 0 [edge case for recursion]
2. height of a tree - using recursion, using global variable, using DFT
    - Using recursion :-
        i. height(tree) = 1 + max (height (left-subtree) , height(right-subtree) )
        ii. height (leaf-node) = 0
    - Using global variable
        i.
    - Using DFT
        i. Apply Depth First Traversal, the highest distance reached from the root node is the height of the tree.
3. how can you traverse a tree without using recursion
    - Using stack and loop (method only shows pre-order traversal, others can be done similarly)
        i. put root node on stack
        ii. while stack is not empty :
        iii. t = stack.pop()
        iv. print val at t
        v. put t->left-subtree on stack [if not null]

vi. put t->right-subtree on stack [if not null]

4. give the level order traversal of a tree - breadth first traversal
5. given a binary tree, transform it into its mirror image.
6. check if the tree is same as its mirror image using only O(1) space.
7. given the preorder and inorder traversals of a tree, give an algorithm to generate the tree.
8. given the preorder and postorder traversals of a tree, a unique tree can't be generated. Why?
9. convert a BST to a sorted linked list, to a sorted doubly linked list
10. reconstruct a BST from a doubly linked list, given the root node
11. given a binary tree, find the largest BST in it
12. find median in a BST in O(log n)
13. find min/max distance between 2 given nodes of a BST
14. find nodes in a tree whose sum is equal to a given number
15. print nodes of a tree height-wise.
16. a tree is represented as a matrix where a(ij) = 1 if j is ancestor of i. Build the tree.
17. All elements of a BST are multiplied by -1. Convert it to a BST once again.
18. given a node, find its nearest sibling

**Linked List**

1. reverse a linked list without using any extra pointer.
2. find the kth last element in a linked list
3. check if there is a loop in a linked list
4. reverse every k nodes of a linked list
5. how to check if a linked list is a palindrome
6. swap kth node from start with the kth node from end in a linked list
7. given 2 linked lists, find if they intersect. find the first common node.
8. merge 2 sorted linked lists

**Stacks and queues**

1. given 2 queues, simulate a stack. given 2 stacks, simulate a queue. what about run time of the operations
2. implement stack for push, pop and min in O(1).
3. how to implement 2 stacks using an array. how to implement 3 ?

**Heaps / Priority Queue**

1. find the kth largest number from a stream of numbers.

**Trie**

1. given a stream of numbers, design a data structure to store the numbers along with their frequencies.
2. given a list of words, find all the anagram pairs.
3. longest common substring between 2 strings
4. longest palindrome in a string.
5. longest repeated substring in a string.
6. first non-repeated character in a string
7. find frequency of all words in a string, all letters in a word, all items in a list

**Others**

1. given a matrix with sorted rows and columns, search for an element in O(log n)
2. find 3 numbers a,b and c in an array such that a+b=c. Does it help if the array is sorted.
3. find largest number from the digits of a given number without extra space
4. given 2n+1 numbers in an array in which n numbers are repeated. find the non-repeated one. given 2n+2, find the 2 non-repeated.
5. reverse a number without using any extra space.