Mid-Term Exam

Time allowed : 120 minutes

Instructions

1. Each question carries 10 points (Total : 40 points)
2. You should include a main method in your classes. Test your code with at-least one case (this should be reflected in the main method). Solutions without a test case for a problem in the main method will not be evaluated.
3. Be sure to handle the possible edge-cases.
4. You should mention the Time & Space complexity for every problem (carries 10% of the grade)
5. The code definitions provided below are in java. If you are using some other language, you can assume a similar definition.
6. Late submissions : -2 points for every 10 minutes.
7. Please do not copy code.

ALL THE BEST !!

Q-1 : Write a function to determine whether two singly-linked lists are converging.

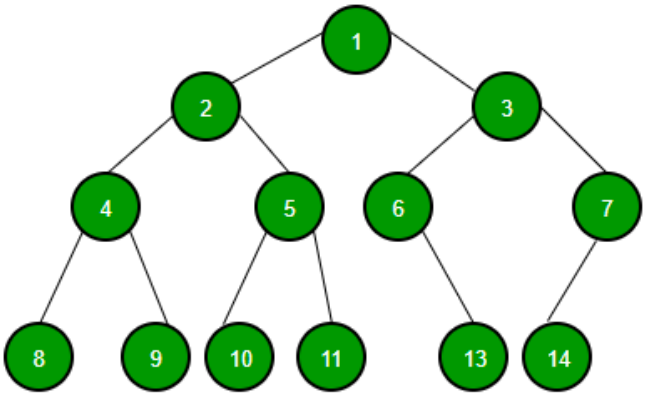
```
class ListNode {
    int val;
    ListNode next;
    ListNode() {}
    ListNode(int val) { this.val = val; }
    ListNode(int val, ListNode next) { this.val = val; this.next = next; }
}

class Solution {
    public boolean areConverging(ListNode n1, ListNode n2){

    }
}
```

Q-2 : Print perimeter of a tree (in clock-wise order).

Example : Expected output for the following tree should be : [1,3,7,14,13,11,10,9,8,4,2]



```
public class TreeNode {
  int val;
  TreeNode left;
  TreeNode right;
  TreeNode() {}
  TreeNode(int val) { this.val = val; }
  TreeNode(int val, TreeNode left, TreeNode right) {
      this.val = val;
      this.left = left;
      this.right = right;
  }

class Solution{
   public void printPerimeter(TreeNode root){

   }
}
```

Q-3 : Given a binary array, find the maximum number of consecutive 1s in this array.

Example : For the given array [0,1,0,1,1,0,1,1,1,0,0,0] the result should be 3.

```
class Solution{
    public int getMaxConsecutiveOnes(int[] nums){

    }
}
```

Q-4 : Given a sorted array 'nums' , return the first occurrence of an integer 'x'.

Example : For the given array [2,4,4,4,6,7,7,7,8,9,9,9,] & x = 7 , the result should be 5.
Please note that you should not use a linear search to solve this problem.

```
class Solution {
    public int getIndex(int[] nums, int x){

    }
}
```