Question 1)

(A)

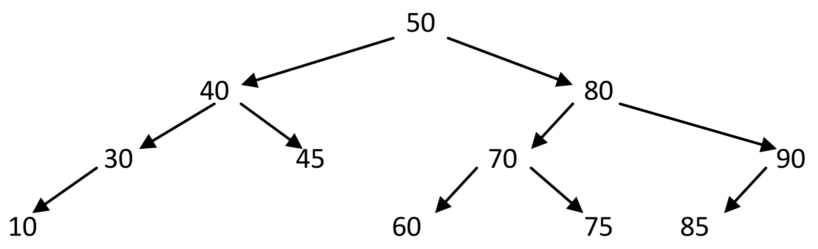
Pre-order : 1-2-4-7-8-5-9-10-12-13-3-6-11

In-order : 7-4-8-2-9-5-12-10-13-1-3-6-11

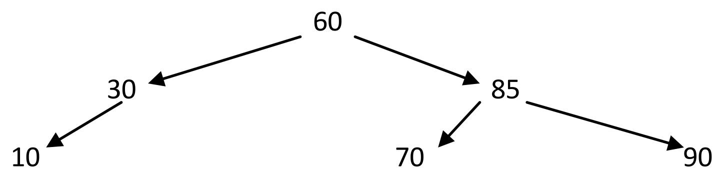
Post-order : 7-8-4-9-12-13-10-5-2-3-6-11-1

(B)

Before delation   Tree



After Delation:



Question 3

The addNgram function is a recursive function and its base case is if root is null, create new node and add it. Because of Binary Search Tree(BST), all left child must be lower (descending order) than its parent and all right child must be greater(ascending order) than its parent. The addition must be applied as obeying these rules. When adding node, firstly we must find true place :

In best case, which is that there is no node ad add it as root, this operation takes θ(1)

In average case, these operations take θ(logn).

In worst case, which is that for adding look at all left child or all right child and in every time look at half of array , these operations take θ(n). Since for finding right place look at every leftmost or rightmost child (it is n traversal) takes θ(n).

The printNgramFrequency function traverses every node and look at its frequency. Because of traversing all tree, it takes θ(n). This case there is no any priority and whether input is sorted or not doesn't matter. In every case its complexity is θ(n).sss