Title: Binary Search Trees
Author: EFE ACER
ID: 21602217
Section: 3
Assignment: 2
Description: The answers of questions 1 and 3 are given in this pdf.

Question 1:

(a)

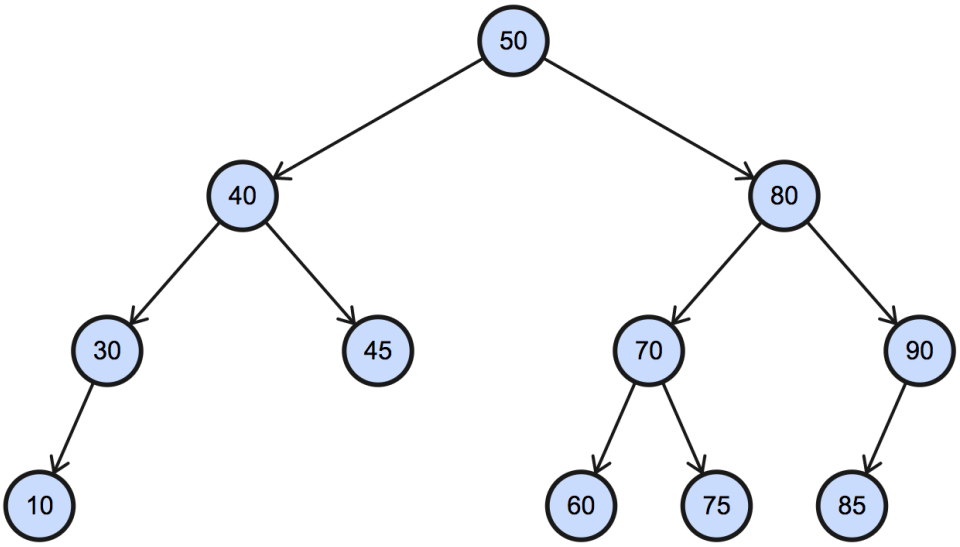
Preorder Traversal: M, O, L, A, G, H, R, T, I

Inorder Traversal: A, L, O, G, M, R, H, T, I

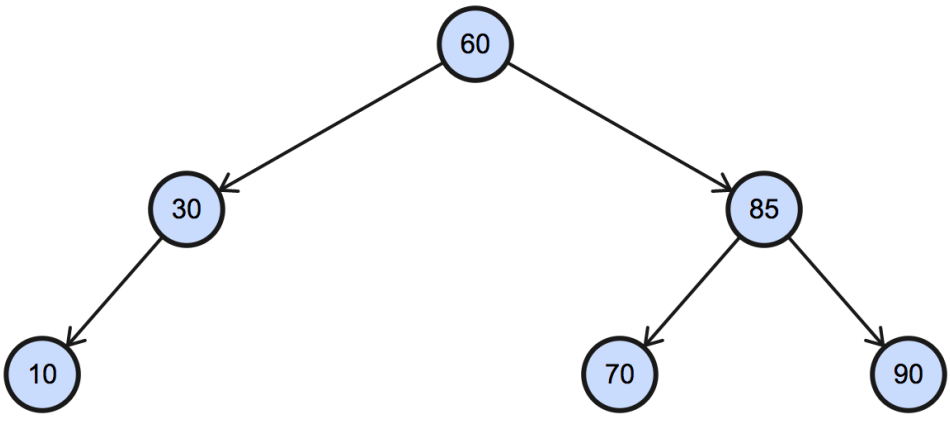
Postorder Traversal: A, L, G, O, R, I, T, H, M

(b)

The final Binary Search Tree after all insertions:

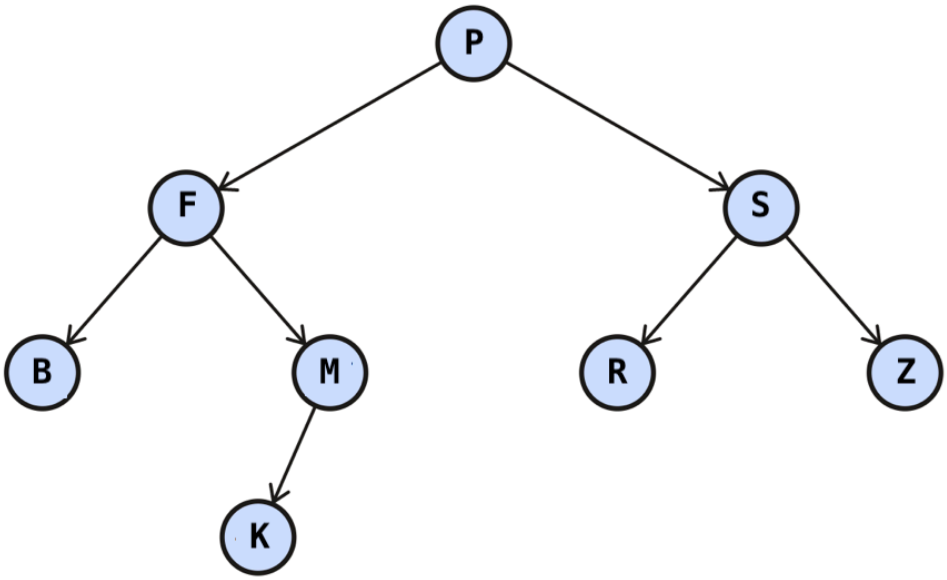


The final Binary Search Tree after all deletions:



(c)

The Binary Search Tree constructed from the given Preorder Traversal:



Postorder Traversal: B, K, M, F, R, Z, S, P

Question 2:

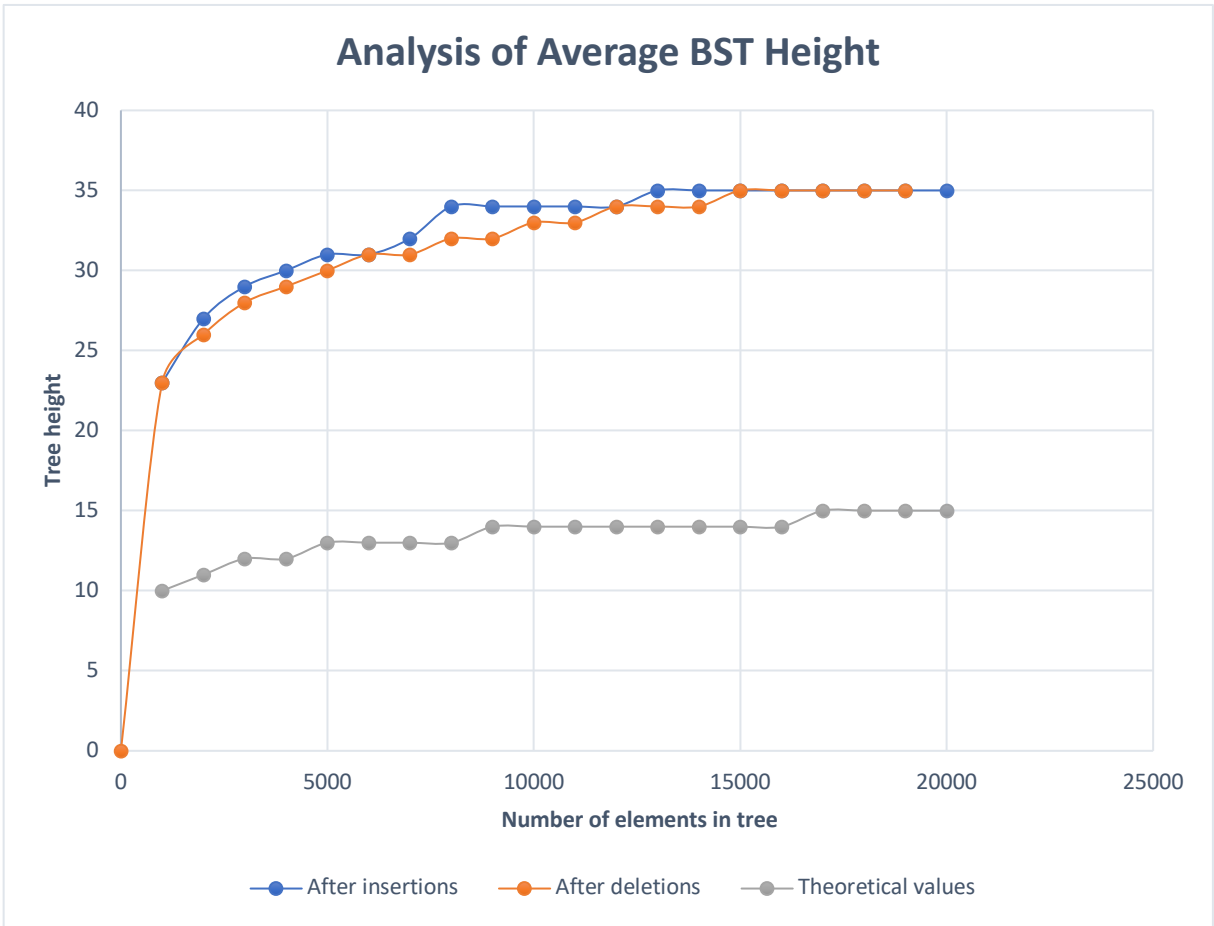


Figure 1

Comments:

- Height of a Binary Search Tree (BST) is, by definition, the number of nodes on the longest path from the root to any leaf. The maximum height (worst-case height) of a BST is apparently n, which can be generalized as O(n) and is the case for a sorted insertion. On the other hand, the minimum height of a BST with n nodes is exactly $\log_2(n+1)$, which is the case for the complete binary tree with n nodes. This result can be generalized so that we can say the best-case height of a BST is order of $\log(n)$, that is O($\log(n)$). Analysis show that as we keep inserting random numbers to a BST, the BST approximately becomes height balanced, that is the height of any node's right subtree and left subtree differ no more than 1. This approximation allows us to state that the average-case height of a BST is also O($\log(n)$). The theoretical values for this particular analysis is given in Figure 1. It is clear that theoretical values form a logarithmic function. The practical results of the experiment came out to be a little off from the theoretical values. However, they are still very close to the theoretical values compared to the worst-case height values. This indicates that the findings of this experiment are in agreement with the theoretical values.
- As I have already stated in the first bullet, the worst-height of a BST is n and this happens for a sorted insertion. The reasoning is that, since every value to the right of a node is by definition greater than itself; only the nodes in the rightmost side of the tree will be filled in the case of a sorted (ascending) insertion. Similarly, in the case of a descending insertion, only the nodes in the leftmost side of the tree will be filled. Both cases will result in a linear, linked-list like structure, where the height and most of the operations become order of n.