Binary Search Tree Examples

Problem 1

- Consider the binary search tree T in Fig. 7.73. Suppose ITEM = 33 is added to the tree T. (a) Find the new tree T.

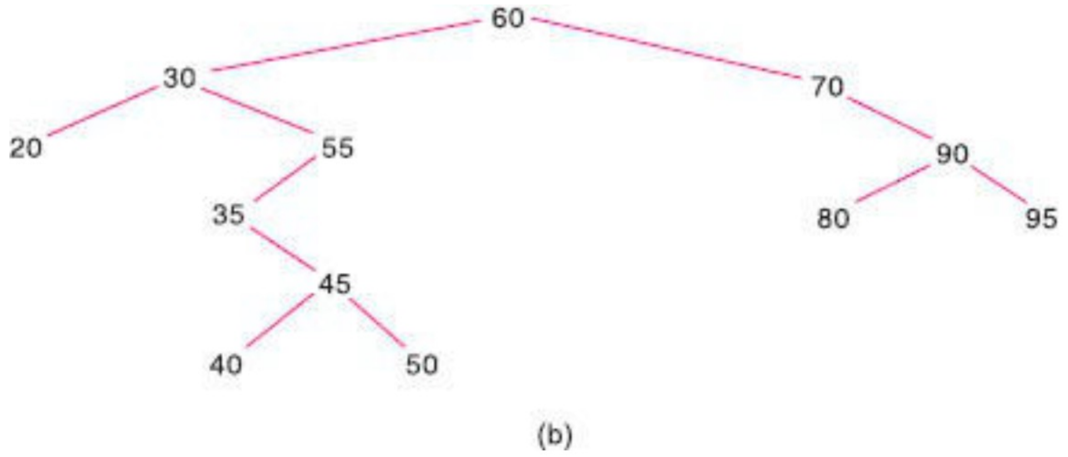
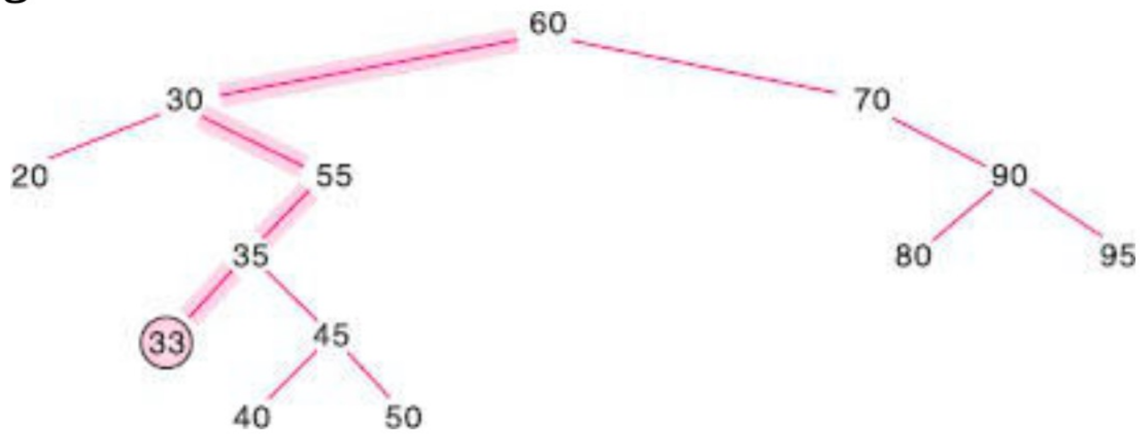


Fig. 7.73

Problem 1: Solution

(a) Compare ITEM = 33 with the root, 60. Since 33 < 60, move to the left child, 30. Since 33 > 30, move to the right child, 55. Since 33 < 55, move to the left child, 35. Now 33 < 35, but 35 has no left child. Hence add ITEM = 33 as a left child of the node 35 to give the tree in Fig. 7.77. The shaded edges indicate the path down through the tree during the insertion algorithm.



Problem 2

Suppose the following list of letters is inserted in order into an empty binary search tree:
J, R, D, G, T, E, M, H, P, A, F, Q
(a) Find the final tree T and (b) find the inorder traversal of T.

Problem 2: Solution

(a) Insert the nodes one after the other to obtain the tree in Fig. 7.79.

(b) The inorder traversal of T follows:

A, D, E, F, G, H, J, M, P, Q, R, T

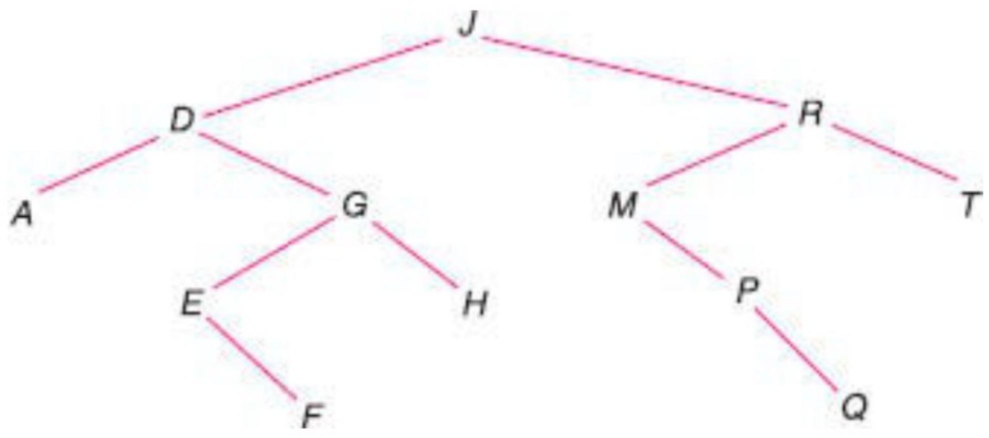


Fig. 7.79

Problem 3

Consider the binary search tree T in Fig. 7.79. Describe the tree after (a) the node M is deleted and (b) the node D is also deleted.

Problem 3: Solution

(a) The node *M* has only one child, *P*. Hence delete *M* and let *P* become the left child of *R* in place of *M*.

(b) The node *D* has two children. Find the inorder successor of *D*, which is the node *E*. First delete *E* from the tree, and then replace *D* by the node *E*.

Figure 7.80 shows the updated tree.

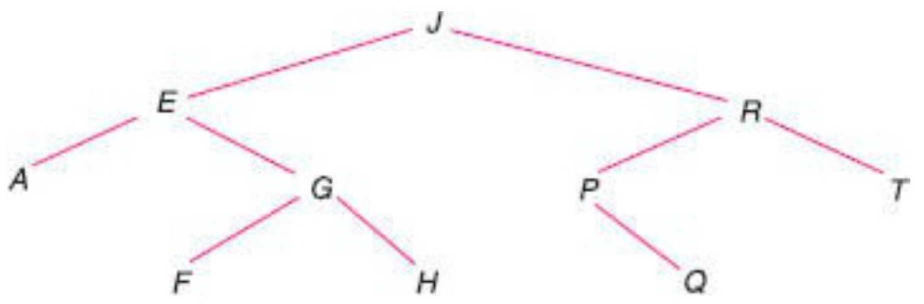


Fig. 7.80

Problem 4

Suppose n data items $A_1, A_2, \ldots,$ AN are already sorted, i.e., $A_1 < A_2 < \ldots < A_N$

(a) Assuming the items are inserted in order into an empty binary search tree, describe the final tree T.
(b) What is the depth D of the tree T?

(a) The tree will consist of one branch which extends to the right, as pictured in Fig. 7.81.
(b) Since T has a branch with all n nodes, D = n.

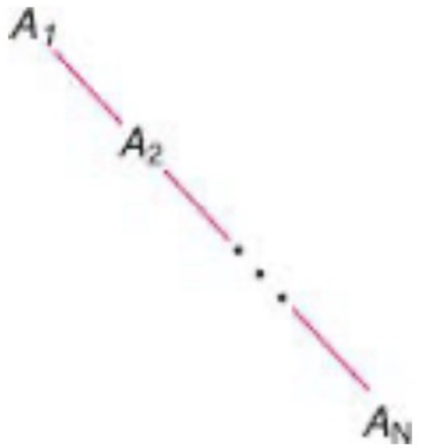


Fig. 7.81