Data Structures

Topic: Binary Search Tree

By

Ravi Kant Sahu

Asst. Professor,

Lovely Professional University, Punjab

Contents

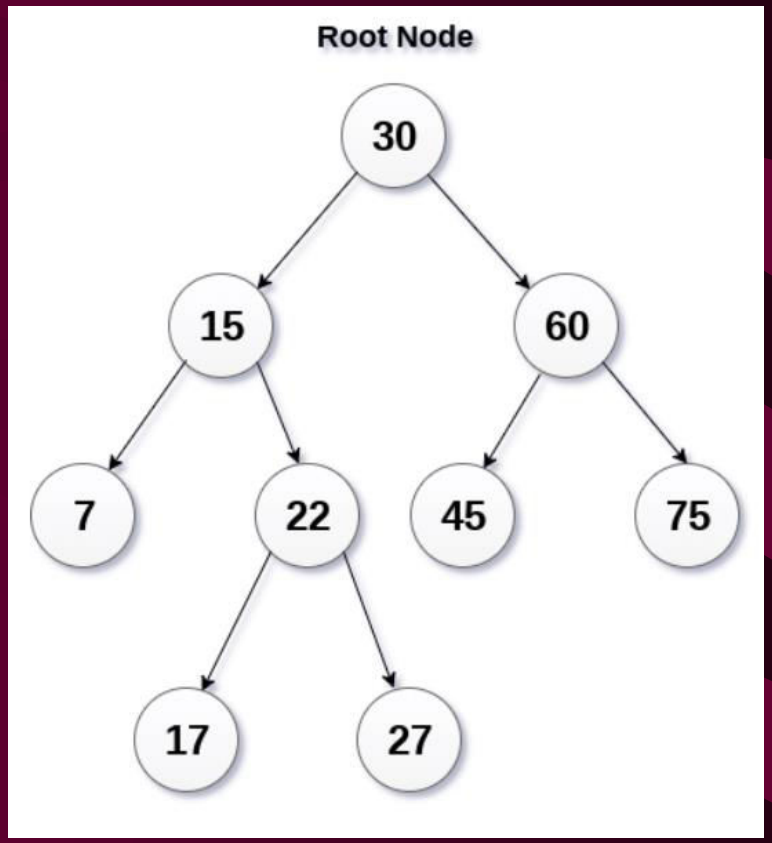
- Introduction
- Searching in BST
- Insertion in BST
- Deletion in BST
- Review Questions

Binary Search Tree

Binary Search Tree is a Binary Tree which has the following properties:

1. The left subtree of a node N contains only nodes with keys lesser than the node's key.

2. The right subtree of a node N contains only nodes with keys greater than the node's key.



Binary Search Tree

Inorder Traversal of a Binary Search Tree always gives the sorted sequence of all the Keys.

It means that if the Keys of any BST is known then Inorder Traversal can be directly obtained by Sorting the Keys in ascending order.

Exercise

Given the Postorder Traversal of a Binary Search Tree:

Postorder: 35, 40, 30, 65, 60, 80, 70, 50

Find the Preorder Traversal of the BST.

Searching in BST

SEARCH_BST(INFO, LEFT, RIGHT, ROOT, ITEM, LOC)

1. Set PTR = ROOT and LOC = NULL.
2. Repeate step 3 while PTR!= NULL:
3. If ITEM = INFO [PTR], then:
   Set LOC = PTR and Exit.
   else If ITEM < INFO[PTR], then:
   Set PTR = LEFT [PTR].
   else
   Set PTR = RIGHT [PTR].
   [End of Loop]
4. If LOC = NULL, then Write “Search unsuccessful”.
5. Return LOC.

Insertion in BST

INS_BST(INFO, LEFT, RIGHT, ROOT, AVAIL, ITEM, LOC)

1. If AVAIL = NULL, then: Write “OVERFLOW” and Exit.
2. Call FIND (INFO, LEFT, RIGHT, ROOT, ITEM, LOC, PAR)
3. If LOC != NULL, then Exit.
4. [Ccreate NEW Node]
   (a) Set NEW = AVAIL, AVAIL = LEFT[AVAIL] and INFO[NEW] = ITEM.
   (b) Set LEFT[NEW] = NULL and RIGHT[NEW] = NULL.
5. [Add ITEM to Tree]
   If PAR = NULL, then
      Set ROOT = NEW.
   Else if ITEM < INFO[PAR], then:
      Set LEFT[PAR] = NEW.
   Else: Set RIGHT [PAR] = NEW.
6. Exit.

Insertion in BST...

FIND (INFO, LEFT, RIGHT, ROOT, ITEM, LOC, PAR)

1. If ROOT = NULL, then:
   Set PAR = NULL and LOC = NULL and Return.
2. Set PTR = ROOT and SAVE = NULL.
3. Repeate while PTR!= NULL
4. If ITEM = INFO[PTR], then
   Set PAR = SAVE and LOC = PTR and Return.

   else if ITEM < INFO [PTR], then:

   Set SAVE = PTR and PTR = LEFT [PTR].

   else

   Set SAVE = PTR and PTR = RIGHT [PTR].

[End of Loop]

6. [Search Unsuccessful]

If PTR= NULL, then Return PAR= SAVE and LOC = NULL.

Exercise

Create a Binary Search Tree from the following elements:

A. 40, 15, 25, 50, 30, 20, 35

B. G, S, E, A, K, J, C

Deletion in BST

Find the location of node N which contains the ITEM.

CASE 1:

N has no children. Then N is deleted from T by simply replacing the location of N in the parent node P(N) by Null Pointer.

CASE 2:

N has exactly one child. Then N is deleted from T by simply replacing the location of N in P(N) by the location of the only child of N.

CASE 3:

N has two children. Let S(N) denote the inorder successor of N. Then N is deleted from T by first deleting S(N) from T and then replacing node N in T by the node S(N).

Deletion in BST

DELETE_BST (INFO, LEFT, RIGHT, ROOT, AVAIL, ITEM)

1. Call FIND (INFO, LEFT, RIGHT, ROOT, ITEM, LOC, PAR)
2. If LOC = NULL, then: Write "ITEM not in Tree" and Exit.
3. If RIGHT[LOC] != NULL and LEFT [LOC] != NULL, then:
   Call DELETE_B (INFO, LEFT, RIGHT, ROOT, LOC, PAR)

   Else:

   Call DELETE_A (INFO, LEFT, RIGHT, ROOT, LOC, PAR))
4. Set LEFT[LOC] = AVAIL and AVAIL = LOC.
5. Exit.

Deletion in BST (1)

DELETE_A(INFO, LEFT, RIGHT, ROOT, LOC, PAR)

1. If LEFT[LOC] = NULL, and RIGHT[LOC] = NULL, then:
   Set CHILD = NULL.
   Else if LEFT[LOC] != NULL, then:
   Set CHILD = LEFT [LOC].
   Else
   Set CHILD = RIGHT [LOC]
2. If PAR != NULL, then:
   If LOC = LEFT[PAR], then:
   Set LEFT[PAR] = CHILD.
   Else: Set RIGHT[PAR] = CHILD.
   Else: Set ROOT = CHILD.
3. Return

Deletion in BST(2)

DELETE_B(INFO, LEFT, RIGHT, ROOT, LOC, PAR)

1. [Find SUC and PARSUC]
   (a) Set PTR = RIGHT[LOC] and SAVE = LOC.
   (b) Repeat while LEFT[PTR] != NULL:
   Set SAVE = PTR and PTR = LEFT [PTR]
   (c) Set SUC = PTR and PARSUC = SAVE.
2. [Delete Inorder Successor]
   Call DELETE_A (INFO, LEFT, RIGHT, ROOT, SUC, PARSUC)
3. [Replace node N by its inorder successor.]
   (a) If PAR != NULL, then:
   If LOC = LEFT[PAR], then:
   Set LEFT[PAR] = SUC
   Else: Set RIGHT [PAR] = SUC
   Else: Set ROOT = SUC.
   (b) Set LEFT[SUC] = LEFT[LOC] and
   RIGHT[SUC] = RIGHT[LOC].
4. Return

Exercise

Given that a Binary Search Tree is represented using Sequential Representation as shown below:

TREE = {L, D, P, B, G, M, _, _, C, F}

Which of the following Key must substitute D, when Deletion of D is performed?

A. G B. F C. B D. Either G or F

Questions

Review Questions

- What is the maximum size of array required to represent a tree with height h, using sequential representation?

- Sequential representation of tree is more efficient than linked. When?