Properties

In this lesson, we'll discuss a few properties of BST due to their structure.

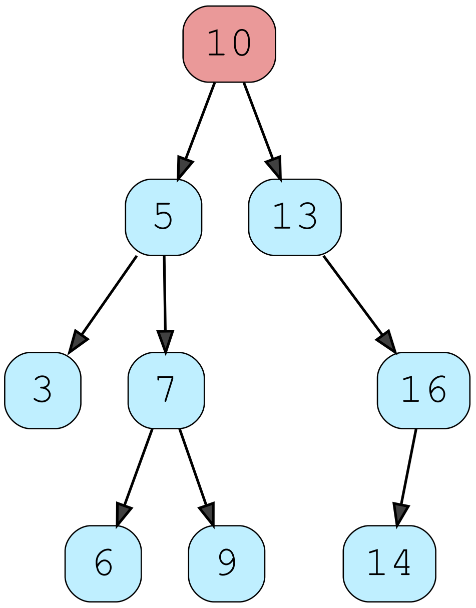
We'll cover the following

- Min value node
- Max value node

Min value node

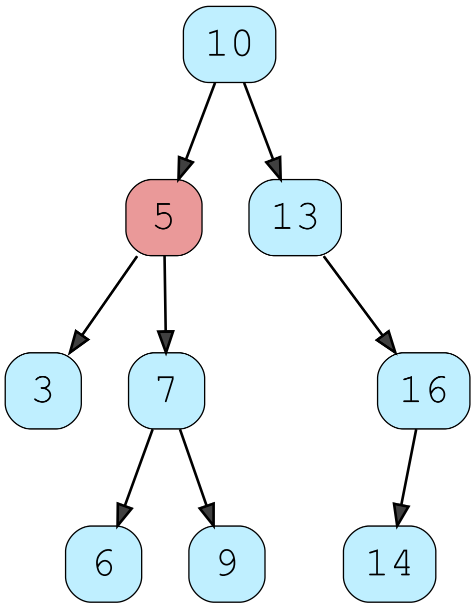
For every node, the left key is smaller, and the right key is greater. If we keep traversing left for each node starting from the root, the leaf we reach is the smallest value node in the entire BST.

Time Complexity: In the case of a skewed tree, the height of the tree is N. The time Complexity to find the minimum node is $O(N)$



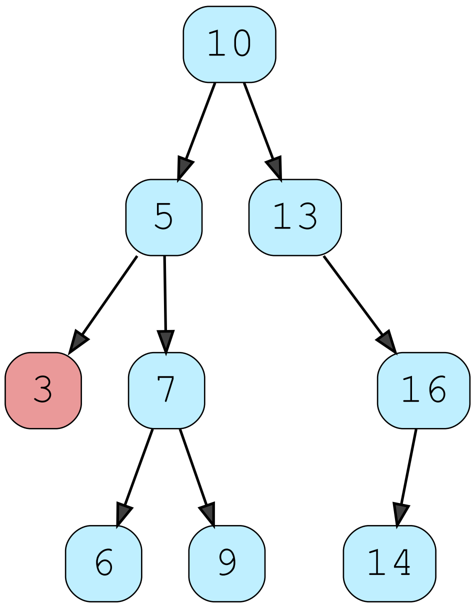
Starting from root, go left

1 of 4



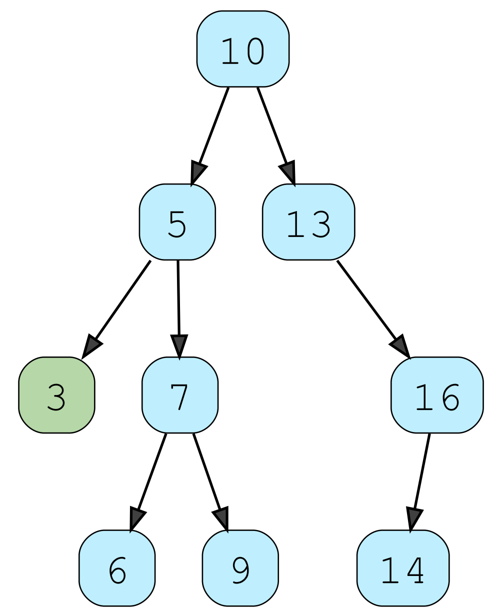
go in the left subtree

2 of 4



Left subtree is null for 3

3 of 4



3 is the minimum value node

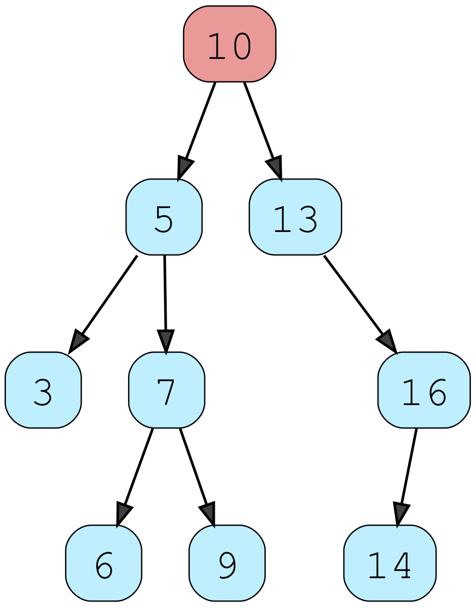
4 of 4

```
struct Node* min_value(struct Node* node) {
    struct Node* pCrawl = node;

    while (pCrawl->left != NULL)
        pCrawl = pCrawl->left;

    return pCrawl;
}
```

Max value node

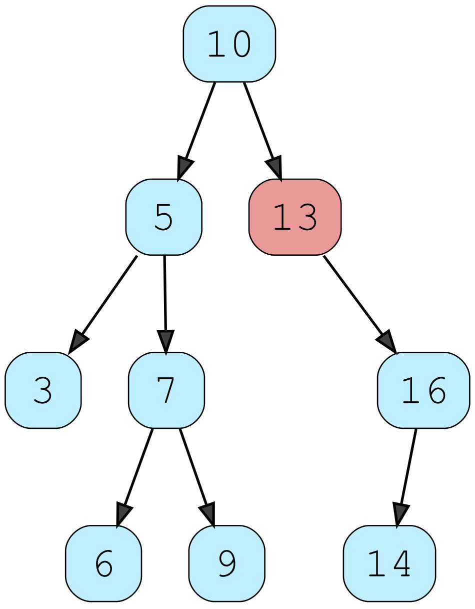
Similarly, we can find the max value node by traversing to the right subtrees.

Time Complexity: $O(N)$



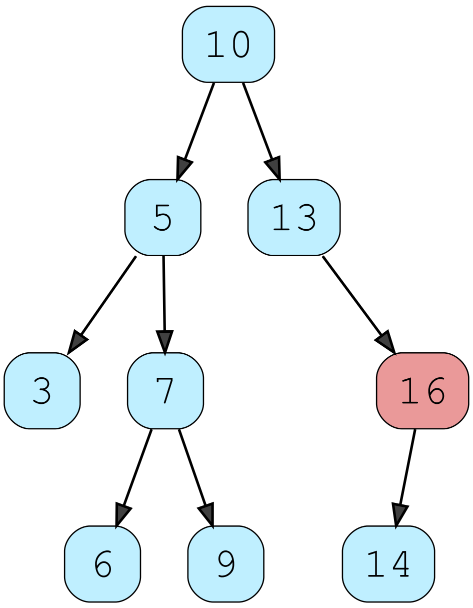
Starting from root, go right

1 of 4



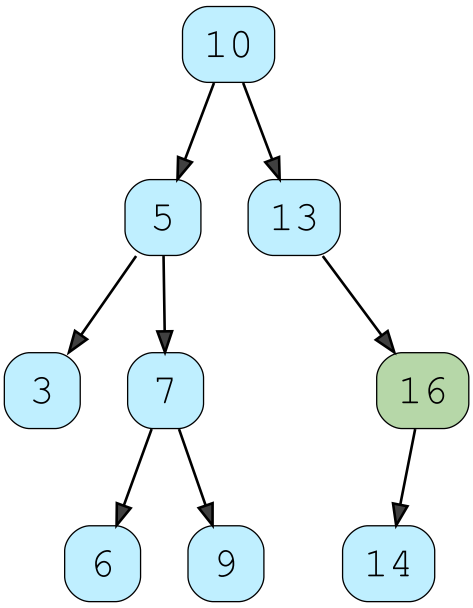
go in the right subtree

2 of 4



right subtree is null

3 of 4



16 is the maximum value node

4 of 4

```
struct Node* min_value(struct Node* node) {
    struct Node* pCrawl = node;

    while (pCrawl->right != NULL)
```

```
        pCrawl = pCrawl->right;



    return pCrawl;
}
```

In the next lesson, we’ll discuss the delete operation on a BST.