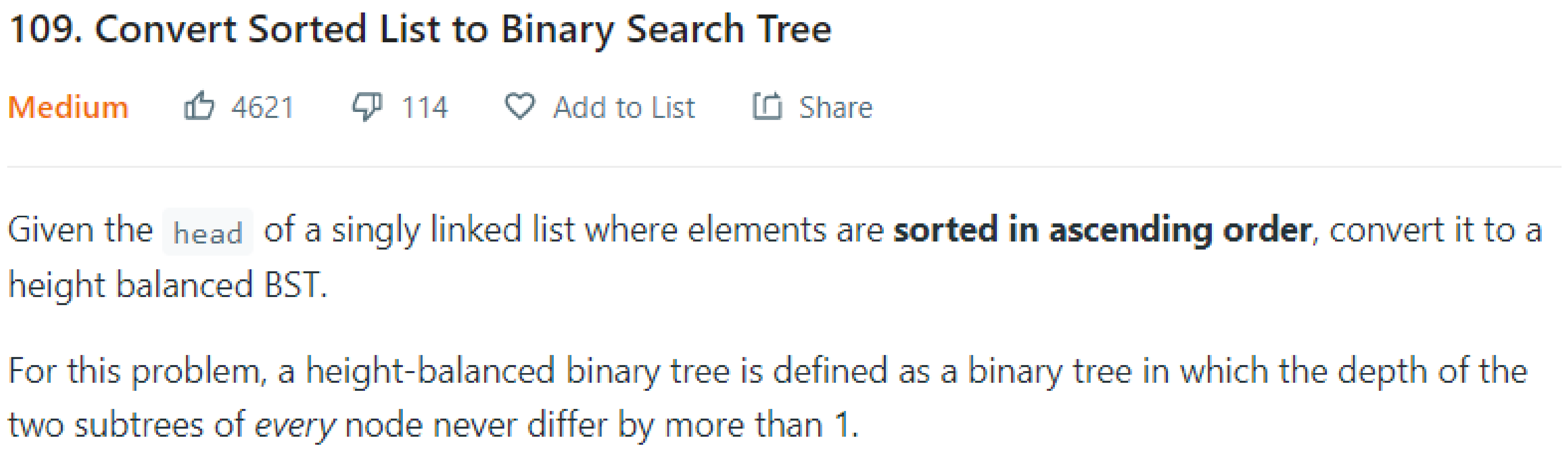
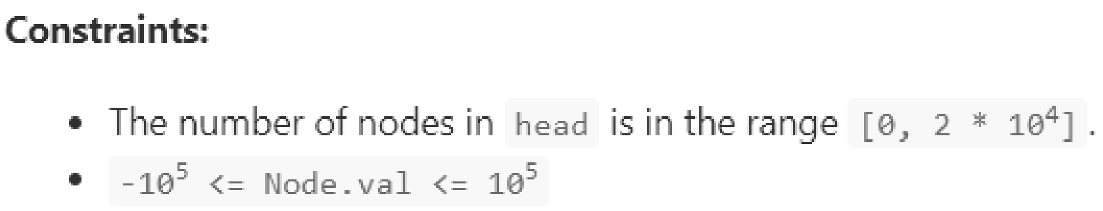


109. Convert Sorted List to Binary Search Tree

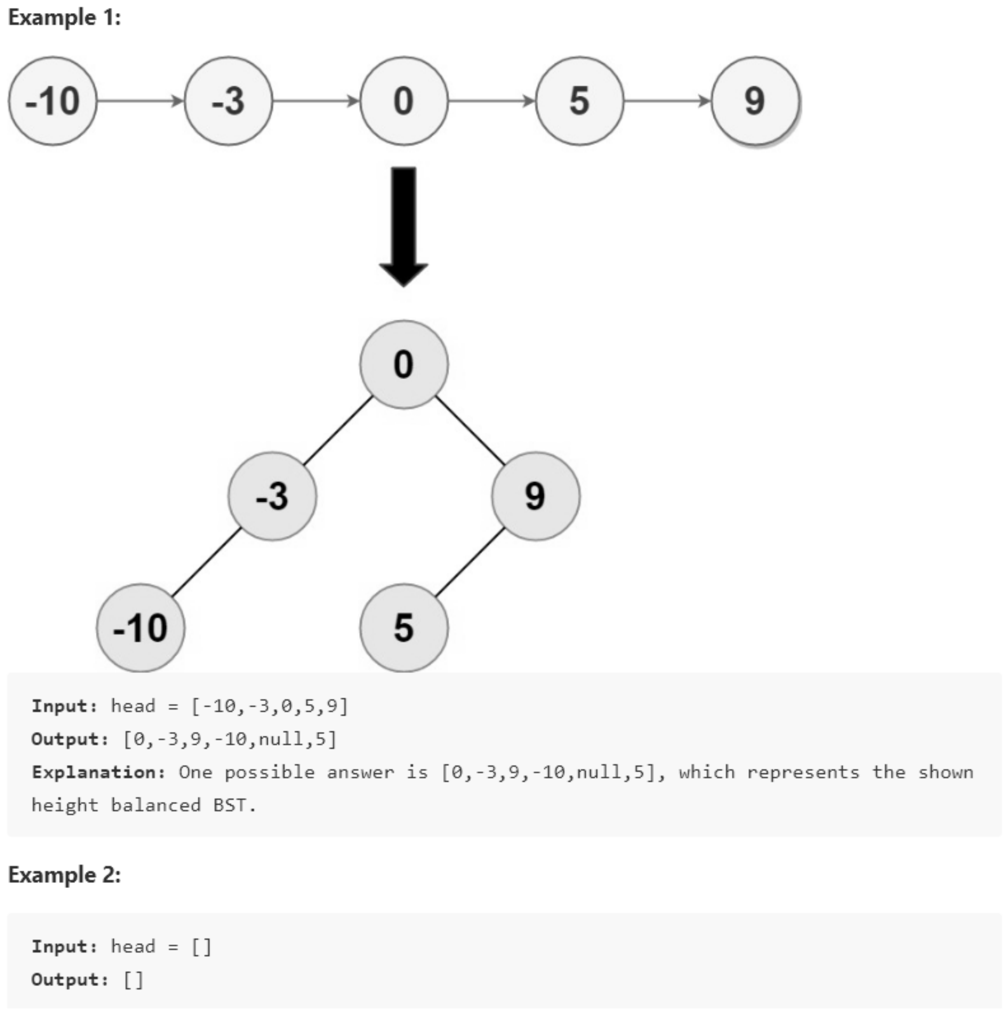
Given the `head` of a singly linked list where elements are **sorted in ascending order**, convert it to a height balanced BST.

For this problem, a height-balanced binary tree is defined as a binary tree in which the depth of the two subtrees of *every* node never differ by more than 1.



Constraints:

- The number of nodes in `head` is in the range `[0, 2 * `10^4`]`.
- -10^5 `<= Node.val <=` 10^5

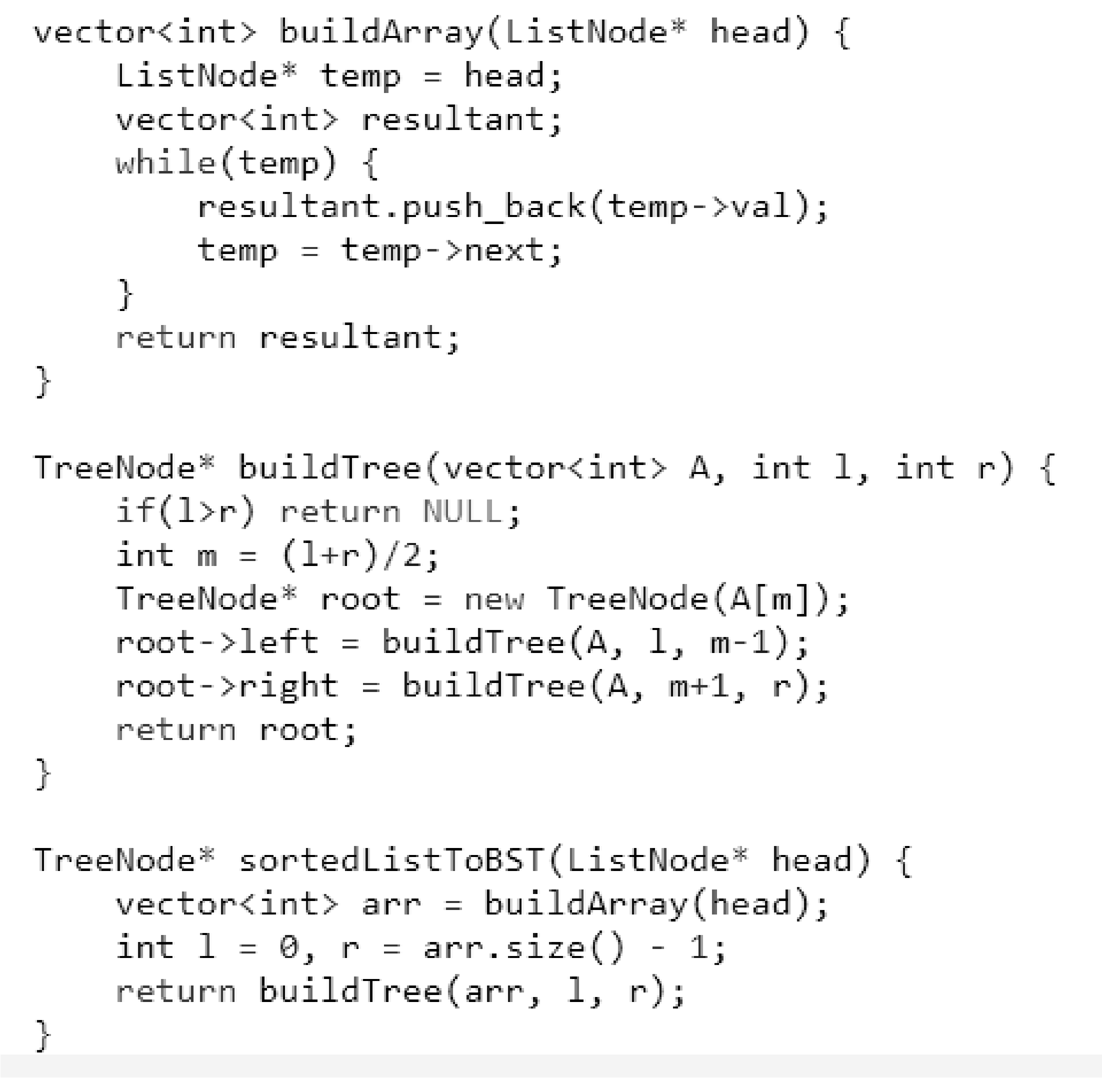


Example 1:

```
Input: head = [-10,-3,0,5,9]
Output: [0,-3,9,-10,null,5]
Explanation: One possible answer is [0,-3,9,-10,null,5], which represents the shown
height balanced BST.
```

Example 2:

```
Input: head = []
Output: []
```



```
vector<int> buildArray(ListNode* head) {
    ListNode* temp = head;
    vector<int> resultant;
    while(temp) {
        resultant.push_back(temp->val);
        temp = temp->next;
    }
    return resultant;
}

TreeNode* buildTree(vector<int> A, int l, int r) {
    if(l>r) return NULL;
    int m = (l+r)/2;
    TreeNode* root = new TreeNode(A[m]);
    root->left = buildTree(A, l, m-1);
    root->right = buildTree(A, m+1, r);
    return root;
}

TreeNode* sortedListToBST(ListNode* head) {
    vector<int> arr = buildArray(head);
    int l = 0, r = arr.size() - 1;
    return buildTree(arr, l, r);
}
```

#100daysofDSA

/rvislive

Rakesh Vishwakarma