	National University of Computer and Emerging Sciences (Lahore)			
	Course:	**Applied Programming**	**Code:**	**CS-0319**
	Section:	**MSCS-2A**	**Semester:**	**Spring 2024**
	Time:	**25 minutes**	**TotalMarks:**	**10**
	Date:		**Roll no:**	
	Name:			

Question#1: **[5]**

Consider the following list of values to be inserted into the Binary Search Tree (BST): 5, 10, 15, 20, 25, 30, 35.

Draw the Binary Search Tree that results from inserting the above list of values in order.

Tree after insertion:

Consider the following piece of C++ code:

```
TreeNode* leftRotate(TreeNode* root) {
    TreeNode* newRoot = root->right;
    root->right = newRoot->left;
    newRoot->left = root;
    return newRoot;
}
TreeNode* RotationThrice(TreeNode* root) {
    for (int i = 0; i < 3; ++i) {root = leftRotate(root);}
    return root;
}
```

Suppose the function `RotationThrice(TreeNode* root)` is called for the root node of the above drawn tree. Redraw the updated tree below:

Updated Tree:

Question#2: **[2+3]**

You are provided with an unordered array of integer values. Your task is to create an integer Binary Search Tree (BST) from this array using an insert function:

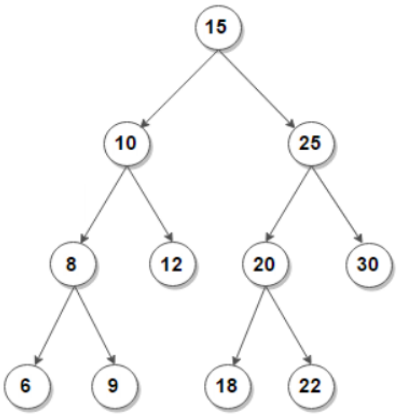
```
1)Node* insertIntoBST(int * arr, int size)
```

Once the BST is constructed, count the number of subtrees within it recursively where all nodes lie within a given range [L, R] (Subtrees here refer to any node and its descendants):

```
2)int countSubtreesInRange(Node* root, int L, int R)
```

NOTE: you may create other helper functions (if needed)

For Example: Consider the following BST. The total number of subtrees with nodes in range [5, 20] is 6.



The subtrees are: