Experiment : 2.2

Student Name: SANJIV GUPTA
Branch: CSE
Semester: 5th
Subject Name: Advanced Programming LAB

UID: 21BCS-*3478*
Section/Group: 21BCS-IOT-602B
Date: 29/09/23
Subject Code: 21CSP-314

AIM:
To implement the concept of Tree.

OBJECTIVE:
1). Given a pointer to the root of a binary tree, print the top view of the binary tree.The tree as seen from the top the nodes, is called the top view of the tree.

2.) You are given a pointer to the root of a binary search tree and values to be inserted into the tree. Insert the values into their appropriate position in the binary search tree and return the root of the updated binary tree. You just have to complete the function.

CODE:

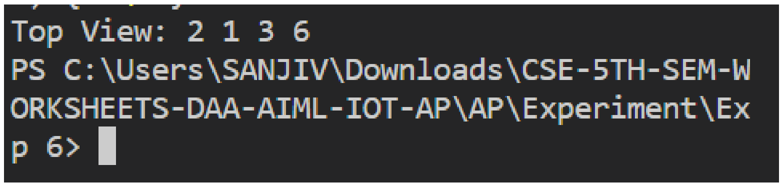
Code 1:
```
#include <iostream>
#include <map>
#include <queue>
using namespace std;

struct TreeNode {
    int data;
    TreeNode* left;
    TreeNode* right;
    TreeNode(int val) : data(val), left(nullptr), right(nullptr) {}
};

void topView(TreeNode* root) {
    if (!root) return;

    map<int, int> horizontalDistanceMap;

    queue<pair<TreeNode*, int>> q;
    q.push({root, 0});

    while (!q.empty()) {
        auto current = q.front();
        q.pop();

        TreeNode* node = current.first;
        int hd = current.second;

        if (horizontalDistanceMap.find(hd) == horizontalDistanceMap.end()) {
            horizontalDistanceMap[hd] = node->data;
        }

        if (node->left) {
            q.push({node->left, hd - 1});
        }

        if (node->right) {
            q.push({node->right, hd + 1});
        }
    }

    for (const auto& entry : horizontalDistanceMap) {
        cout << entry.second << " ";
    }
    cout << endl;
}

int main() {
    TreeNode* root = new TreeNode(1);
    root->left = new TreeNode(2);
    root->right = new TreeNode(3);
    root->left->right = new TreeNode(4);
    root->left->right->right = new TreeNode(5);
    root->left->right->right->right = new TreeNode(6);

    cout << "Top View: ";
    topView(root);

    return 0;
}
```

Code 2

```
#include <iostream>
using namespace std;

struct TreeNode {
    int val;
    TreeNode* left;
    TreeNode* right;
    TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}
};

TreeNode* insertIntoBST(TreeNode* root, int val) {
    if (!root) {
        return new TreeNode(val);
    }

    if (val < root->val) {
        root->left = insertIntoBST(root->left, val);
    }
    else {
        root->right = insertIntoBST(root->right, val);
    }

    return root;
}

void inorderTraversal(TreeNode* root) {
    if (!root) return;
    inorderTraversal(root->left);
    cout << root->val << " ";
    inorderTraversal(root->right);
}

int main() {
    TreeNode* root = nullptr;
    int values[] = {5, 2, 8, 1, 3, 7, 9};

    for (int val : values) {
        root = insertIntoBST(root, val);
    }

    cout << "Inorder Traversal: ";
    inorderTraversal(root);
    cout << std::endl;
    return 0;
}
```

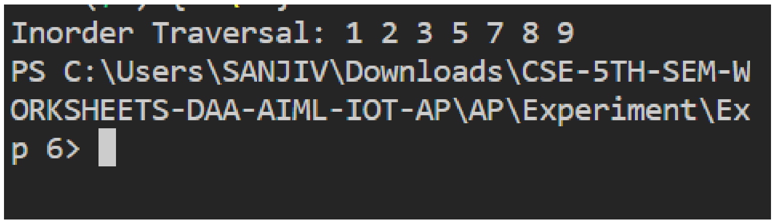
OUTPUT:

OUTPUT 1



```
Top View: 2 1 3 6
PS C:\Users\SANJIV\Downloads\CSE-5TH-SEM-W
ORKSHEETS-DAA-AIML-IOT-AP\AP\Experiment\Ex
p 6>
```

OUTPUT 2



```
Inorder Traversal: 1 2 3 5 7 8 9
PS C:\Users\SANJIV\Downloads\CSE-5TH-SEM-W
ORKSHEETS-DAA-AIML-IOT-AP\AP\Experiment\Ex
p 6>
```

LEARNING OUTCOMES:

1. *Understood the concept of Tree.*
2. *Understood the concept how to search in tree and perform different operations.*
3. *Learn about algorithm thinking*
4. *Learn about mathematical logic*