Rajalakshmi Engineering College

Name: VISHVA RAGAVAN P
Email: 241501249@rajalakshmi.edu.in
Roll no:
Phone: 6381117385
Branch: REC
Department: I AI & ML FC
Batch: 2028
Degree: B.E - AI & ML

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 5_COD_Question 5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

In his computer science class, John is learning about Binary Search Trees (BST). He wants to build a BST and find the maximum value in the tree.

Help him by writing a program to insert nodes into a BST and find the maximum value in the tree.

Input Format

The first line of input consists of an integer N, representing the number of nodes in the BST.

The second line consists of N space-separated integers, representing the values of the nodes to insert into the BST.

Output Format

The output prints the maximum value in the BST.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
10 5 15 2 7

Output: 15

Answer

```
#include <stdio.h>
#include <stdlib.h>

struct TreeNode {
   int data;
   struct TreeNode* left;
   struct TreeNode* right;
};
struct TreeNode* createNode(int key) {
   struct TreeNode* newNode = (struct TreeNode*)malloc(sizeof(struct
TreeNode));
   newNode->data = key;
   newNode->left = newNode->right = NULL;
   return newNode;
}
```

```
// You are using GCC
#include <iostream>
using namespace std;

struct Node {
   int data;
   Node* left;
   Node* right;

   Node(int value) : data(value), left(nullptr), right(nullptr) {}
};

class BST {
```

```
public:
    Node* root;

    BST() : root(nullptr) {}

    void insert(int data) {
        root = insertNode(root, data);
    }

    int findMaxValue() {
        return findMax(root);
    }

private:
    Node* insertNode(Node* node, int data) {
        if (node == nullptr) {
            return new Node(data);
        }
        if (data < node->data) {
            node->left = insertNode(node->left, data);
        } else {
            node->right = insertNode(node->right, data);
        }
        return node;
    }

    int findMax(Node* node) {
        while (node && node->right != nullptr) {
            node = node->right;
        }
        return node->data;
    }
};

int main() {
    int N;
    cin >> N;
    BST tree;

    for (int i = 0; i < N; i++) {
        int value;
        cin >> value;
```

```
        tree.insert(value);
    }

    cout << tree.findMaxValue() << endl;

    return 0;
}
int main() {
    int N, rootValue;
    scanf("%d", &N);

    struct TreeNode* root = NULL;

    for (int i = 0; i < N; i++) {
        int key;
        scanf("%d", &key);
        if (i == 0) rootValue = key;
        root = insert(root, key);
    }

    int maxVal = findMax(root);
    if (maxVal != -1) {
        printf("%d", maxVal);
    }

    return 0;
}
```

Status : Correct ***Marks : 10/10***