Rajalakshmi Engineering College

Name: suriya SM
Email: 241801284@rajalakshmi.edu.in
Roll no: 241801284
Phone: 8110855156
Branch: REC
Department: I AI & DS FD
Batch: 2028
Degree: B.E - AI & DS

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

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

Node* createNode(int data) {
   Node* newNode = (Node*)malloc(sizeof(Node));
   newNode->data = data;
```

```
    newNode->left = newNode->right = NULL;
    return newNode;
}

Node* insert(Node* root, int data) {
    if (root == NULL) return createNode(data);
    if (data < root->data) root->left = insert(root->left, data);
    else root->right = insert(root->right, data);
    return root;
}

int findMax(Node* root) {
    while (root->right != NULL) root = root->right;
    return root->data;
}

int main() {
    int N;
    scanf("%d", &N);
    Node* root = NULL;
    for (int i = 0; i < N; i++) {
        int data;
        scanf("%d", &data);
        root = insert(root, data);
    }
    printf("%d\n", findMax(root));
    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***