[DSA25 End] Max sum path

Submit solution

All submissions
Best submissions

Points: 100 (partial)
Time limit: 1.0s
Memory limit: 256M

Author:
2020111017

Problem type

Allowed languages
C

You are given a binary tree with n nodes, where the root node is 1. The tree structure is described by two arrays:

- Parent Array (parent): An array of size n where parent[i] (0 based) represents the parent of node i+1. For the root node, parent[0] is -1.
- Value Array (value): An array of size n where value[i] (0 based) represents the value of node i+1.

Your task is to build the tree using the given parent array and calculate the maximum path sum from the root node (1) to any other node (can be root also) in the tree.

Input Format

- The first line contains an integer `t`, the number of testcases.
- The second line contains an integer `n`, the number of nodes in the tree.
- The third line contains n space-separated integers, where the i-th integer represents the parent of the i-th node. If the integer is -1, the i-th node is the root node.
- The fourth line contains the value array of size n.

Output Format

- For each test case, a single integer, the maximum path sum from the root node to any other node in the tree.

Constraints

- -1 ≤ parent[i] ≤ n and parent[i] ≠ 0
- $-10^9 \leq val[i] \leq 10^9$

Example

Input

Copy

```
3
5
-1 1 1 2 2
1 2 3 -1 4
4
-1 1 1 2
-10 -10 -10 -10
6
-1 1 1 2 4 4
10 20 30 -100 1000 30
```

Output

Copy

```
7
-10
930
```

Following is the helper code for taking input and building the tree:

Copy

```
    int data;
    struct Node* left;
    struct Node* right;
};

// Function to create a new Node
struct Node* newNode(int x) {
    struct Node* node = (struct Node*)malloc(sizeof(struct Node));
    node->data = x;
    node->left = NULL;
    node->right = NULL;
    return node;
}

-----------------------------------------------------

int n;
scanf("%d", &n);  // Read the number of nodes

int parent[n+1];
struct Node* nodes[n+1];  // Array to store all nodes

// Read the parent array
for (int i = 1; i <= n; i++) {
    scanf("%d", &parent[i]);
    nodes[i] = newNode(i);  // Create a node for each index
}

struct Node* root = NULL;

// Construct the binary tree from the parent array
for (int i = 1; i <= n; i++) {
    if (parent[i] == -1) {
        root = nodes[i];  // Root node has no parent
    } else {
        struct Node* parentNode = nodes[parent[i]];
        if (parentNode->left == NULL) {
            parentNode->left = nodes[i];  // Assign left child
        } else if (parentNode->right == NULL) {
            parentNode->right = nodes[i];  // Assign right child
        }
    }
}
```

Hello, **2024101067**.

No clarifications have been made at this time.