Hello, **2024101067**.

✓ Diameter of a binary tree

Submit solution

My submissions
All submissions
Best submissions

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

Allowed languages
C

Given an array of integers which represents the `parent` of each node of the binary tree, return the length of the `diameter` of the tree.

The `diameter` of a binary tree is the length of the longest path between any two nodes in a tree.

The length of a path between two nodes is represented by the number of edges between them.

You are given an array which describes a tree structure where each element represents the parent of a corresponding node. If parent[i] = -1, node i is the root of the tree and j is the parent of i if parent[i] = j. Moreover it is guaranteed that node 1 will always be the root node and each node will have atmost 2 children.

The tree consists of n nodes, labeled from 1 to n. It is guaranteed that the tree is connected, meaning there is a path between any two nodes, and that there are no self-loops, i.e., for every node i, the parent of i is not i.

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.

Output Format

- For each test case, output an integer representing the diameter of the corresponding binary tree.

Constraints

Hello, **2024101067**.

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

Copy

Hello, **2024101067**.

```
    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
        }
    }
}
```

Example

Hello, **2024101067**.

```
10
3
-1 3 1
7
-1 1 5 7 7 4 1
4
-1 3 1 3
1
-1
5
-1 3 4 5 1
1
-1
1
-1
6
-1 4 2 1 3 2
1
-1
4
-1 4 1 1
```

Copy

Output

```
2
4
2
0
4
0
0
4
0
3
```

Copy

Clarifications

Request clarification

No clarifications have been made at this time.