Alice and the Lonely Nodes

Submit solution

All submissions
Best submissions

Points: 100 (partial)
Time limit: 2.0s
Memory limit: 256M

Allowed languages
C

Alice loves exploring **magical forests**, where trees grow in a special **binary search tree** pattern. In this forest, each tree has **a root** and branches out into smaller nodes.

One day, while observing a **tree**, Alice noticed something interesting—some nodes were **lonely**. A node is **lonely** if:

- It is the **root** (since it has no parent).
- Its parent has **only one child**, meaning it has no siblings.

Alice wants to list all these lonely nodes **in ascending order**, so she can understand how they are spread across the tree.

Can you help Alice find all the lonely nodes in the given **binary search tree (BST)**?

Input Format

- The first line contains an integer **n**, the number of nodes in the BST.
- The second line contains **n** space-separated integers representing the **parent array**, where the **i-th** number represents the **parent of node i** (or **-1** if it is the root).

Output Format

Print all the lonely nodes in **ascending order** in a single line, separated by spaces.

- The **parent array** is given in **0-based indexing**, where **parent[i] = -1** indicates that node **i** is the root.

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];
struct Node* nodes[n];  // Array to store all nodes

// Read the parent array
for (int i = 0; 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 = 0; i < n; i++) {
    if (parent[i] == -1) {
        root = nodes[i];  // Root node has no parent
    } else {
        struct Node* parentNode = nodes[parent[i]];
        if (i<parent[i]) {
            parentNode->left = nodes[i];  // Assign left child
        } else  {
            parentNode->right = nodes[i];  // Assign right child
        }
    }
}
```

Sample Input

```
6
1 -1 3 4 1 4
```

Sample Output

```
1 2
```

Copy

Sample Explanation

The tree structure based on the given **parent array**:

```
  1
 / \
0   4
   / \
  3   5
 /
2
```

Copy

- **Lonely Nodes**:
  - **1** is the **root**, so it is lonely.
  - **2** has no siblings, so it is lonely.

Thus, the output is **1 2** in **ascending order**.

Clarifications

Request clarification

No clarifications have been made at this time.