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

#define max(x, y) ((x) > (y) ? (x) : (y))

typedef struct _node {
    int key;
    struct _node *left;
    struct _node *right;
} node;

// Inserimento in un albero binario non bilanciato.
node *insert(node *root, int key) {
    if (root == NULL) {
        node *new = malloc(sizeof(node));
        new->key = key;
        new->left = NULL;
        new->right = NULL;
        return new;
    }

    if (key <= root->key) {
        root->left = insert(root->left, key);
    } else {
        root->right = insert(root->right, key);
    }

    return root;
}

// Cerca la chiave key all'interno dell'albero e ne restituisce la profondità
int search(const node *root, int key) {
    if (root == NULL) {
        return -1;
    }

    if (root->key == key) {
        return 0;
    }

    int res;
    if (key <= root->key) {
        res = search(root->left, key);
    } else {
        res = search(root->right, key);
    }
    // Se la chiave non è stata trovata nei sottoalberi, dobbiamo propagare -1
    // altrimenti dobbiamo incrementare la profondità di uno.
    return res == -1 ? -1 : 1 + res;
}

int main(int argc, char *argv[]) {
    int k, n, d;
    node *root = NULL;
```

```
    scanf("%d", &n);
    for (size_t i = 0; i < n; i++) {
        scanf("%d", &k);
        root = insert(root, k);
    }

    while (1) {
        scanf("%d", &k);

        if (k < 0) {
            break;
        }

        d = search(root, k);
        if (d == -1) {
            printf("NO\n");
        } else {
            printf("%d\n", d);
        }
    }

    return 0;
}
```