# ВШЭ АиСД 2021. Сбалансированные деревья

4 дек 2021, 18:44:06
старт: 29 окт 2021, 12:00:00
финиш: 9 ноя 2021, 00:00:01
длительность: 10д. 12ч.
начало: 29 окт 2021, 12:00:00
конец: 9 ноя 2021, 00:00:01

## C. Частоты появления элементов (0.25)

| | |
|---|---|
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64.0 Мб |
| Ввод | стандартный ввод |
| Вывод | стандартный вывод |

По данной последовательности постройте дерево, запоминая для каждого элемента его значение и количество его повторений в последовательности.

### Формат ввода

На вход программа получает последовательность целых чисел (пусть $n$ - длина последовательности, $0 \le n \le 2 \cdot 10^6$; числа $\in [-10^9; 10^9]$). Последовательность завершается числом $0$ (оно не входит в последовательность). Постройте дерево, соответствующее данной последовательности.

### Формат вывода

Выведите на экран содержимое дерева в порядке возрастания, по одному элементу на строку. В каждой строке выводите значение элемента, затем, через пробел, укажите, сколько раз он встречается в исходной последовательности.

### Система оценки

| Группа | Баллы | Доп. ограничения | Необх. группы | Комментарий |
|---|---|---|---|---|
| | | $n$ | | |
| 0 | 1 | – | – | Тесты из условия. |
| 1 | 0,5 | $n \le 50$ | 0 | |
| 2 | 1 | $n \le 10^3$ | 0 | |
| 3 | 1,5 | $n \le 10^4$ | 0 | |
| 4 | 2 | $n \le 10^5$ | 0 | |
| 5 | 1 | $n \le 2 \cdot 10^5$ | 0 – 4 | **Offline-проверка** |
| 6 | 1 | $n \le 4 \cdot 10^5$ | 0 – 5 | **Offline-проверка** |

### Пример

**Ввод** **Вывод**

| Ввод | Вывод |
| --- | --- |
| 7 3 2 1 9 5 4 6 8 0 | 1 1<br>2 1<br>3 1<br>4 1<br>5 1<br>6 1<br>7 1<br>8 1<br>9 1 |

Набрать здесь | Отправить файл

```cpp
#include <iostream>

struct TreeNode {
    int value;
    int count;
    TreeNode *leftChildren;
    TreeNode *rightChildren;

    explicit TreeNode(int new_value) {
        value = new_value;
        count = 1;
        leftChildren = nullptr;
        rightChildren = nullptr;
    }
};

class Tree {
private:
    TreeNode *root_;

    TreeNode *insertChildren(TreeNode *tree_node, int value) {
        if (tree_node == nullptr) {
            tree_node = new TreeNode(value);
        } else if (value < tree_node->value) {
            tree_node->leftChildren = insertChildren(tree_node->leftChildren, value);
        } else if (value > tree_node->value) {
            tree_node->rightChildren = insertChildren(tree_node->rightChildren, value);
        } else if (value == tree_node->value) {
            tree_node->count += 1;
        }

        return tree_node;
    }

    void deleteNode(TreeNode *tree_node) {
        if (tree_node) {
            deleteNode(tree_node->leftChildren);
            deleteNode(tree_node->rightChildren);
```

Отправить

Предыдущая | Следующая