



Attacking Vigorously the Leaderboard

locked

 by [piggov](#)

Problem

Submissions

Leaderboard

Discussions

Довършете балансираното AVL дърво като имплементирате следните операции

1. добавяне на елемент
2. премахване на елемент

Забранено е използването на `std::map`

Input Format

При `add` ако числото вече съществува, да изписва `"X already added"` и нов ред след това (на мястото на `X` да се изписва самото подадено число)

При `remove` ако числото не съществува, да изписва `"X not found to remove"` и нов ред след това (на мястото на `X` да се изписва самото подадено число)

Constraints

 $1 \leq N \leq 100\,000$ $\text{int.MIN_VALUE} \leq \text{number} \leq \text{int.MAX_VALUE}$

Output Format

При операция `contains` изпишете `"yes"` или `"no"` в зависимост от това дали даденото число се съдържа в дървото.

При операция `print` изпишете текущото състояние на дървото във формат Ляво-Корен-Дясно с разстояние между елементите.

Note! `cout << fixed`; винаги връща до 6 символа след десетичната запетая.

Sample Input 0

```
7
add 58
add 98
contains 58
add 52
contains 23
add 23
print
```

Sample Output 0

```
yes
no
23.000000 52.000000 58.000000 98.000000
```

Sample Input 1

```

16
add 8.43
add 5.83
add 7.66
add 1.92
remove 7.66
add 4.47
add -2.76
contains 7.23
add -1.64
remove 5.49
add 4.66
add 3.04
add 4.47
contains 8.43
add 7.34
print

```

Sample Output 1

```

no
5.490000 not found to remove
4.470000 already added
yes
-2.760000 -1.640000 1.920000 3.040000 4.470000 4.660000 5.830000 7.340000 8.430000

```

[f](#) [t](#) [in](#)
Submissions: [142](#)



Max Score: 10



Difficulty: Medium

Rate This Challenge:

☆☆☆☆☆

[More](#)

Current Buffer (saved locally, editable)  

C++  

```

1
2
3 ▼ #include <iostream>
4 #include <string>
5 #include <iomanip>
6
7 using namespace std;
8
9 struct Node
10 ▼ {
11     double value;
12     Node *left;
13     Node *right;
14
15     Node(double value, Node *left, Node *right)
16     {
17         this->value = value;
18         this->left = left;
19         this->right = right;
20     }
21 };
22
23 class AVLTree
24 ▼ {
25 private:
26     Node *root;
27
28     bool containsRecursive(Node *current, double value)
29     {
30         if (current == NULL)

```

```
31     {
32         return false;
33     }
34
35     if (current->value == value)
36     {
37         return true;
38     }
39
40     if (value < current->value)
41     {
42         return containsRecursive(current->left, value);
43     }
44     else
45     {
46         return containsRecursive(current->right, value);
47     }
48 }
49
50 void printRecursive(Node *current)
51 {
52     if (current == NULL)
53     {
54         return;
55     }
56
57     printRecursive(current->left);
58     cout << current->value << " ";
59     printRecursive(current->right);
60 }
61
62 public:
63     AVLTree()
64     {
65         root = NULL;
66     }
67
68     void add(double value)
69     {
70
71     }
72
73     void remove(double value)
74     {
75
76     }
77
78     bool contains(double value)
79     {
80         if (root == NULL)
81         {
82             return false;
83         }
84
85         return containsRecursive(root, value);
86     }
87
88     void print()
89     {
90         if (root == NULL)
91         {
92             return;
93         }
94
95         printRecursive(root);
96         cout << endl;
97     }
98 };
99
100 int main()
101 {
102     AVLTree tree;
103     string operation;
104     double number;
```

```
105     int N;
106
107     cin >> N;
108     cout << fixed;
109
110     for (size_t i = 0; i < N; i++)
111     {
112         cin >> operation;
113         if (operation != "print")
114         {
115             cin >> number;
116         }
117
118         if (operation == "add")
119         {
120             tree.add(number);
121         }
122         else if (operation == "remove")
123         {
124             tree.remove(number);
125         }
126         else if (operation == "contains")
127         {
128             if (tree.contains(number))
129             {
130                 cout << "yes" << endl;
131             }
132             else
133             {
134                 cout << "no" << endl;
135             }
136         }
137         else if (operation == "print")
138         {
139             tree.print();
140         }
141     }
142
143     return 0;
144 }
```

Line: 7 Col: 21

 [Upload Code as File](#) ☐ [Test against custom input](#)[Run Code](#)[Submit Code](#)