CS69201: Computing Lab-1
Assignment 4 - AVL Tree
August 14, 2024

==================== Instructions====================

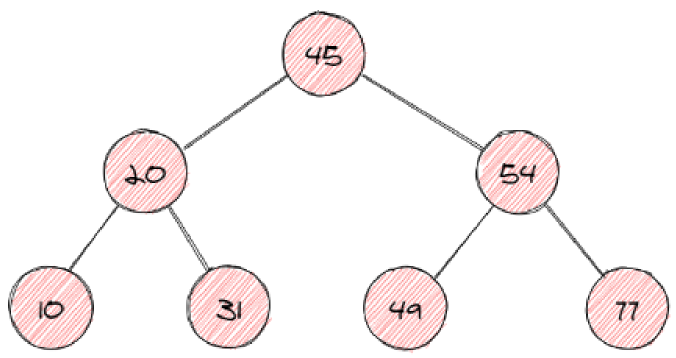
1. In the case of user input assume only valid values will be passed as input.
2. Submit Question 1 by 5 PM
3. You can use C or C++ as the programming language. **However, you are not allowed to use any STL libraries in C++**
4. Regarding Submission: For each question create a separate C file. -> <rollno>_Q1.c, <rollno>_Q2.c, <rollno>_Q3.c. Create a zip file of all these C files in the name <rollno>_A4.zip and submit it to Moodle. For example, if your roll number is 24CS60R15, then your file name will be 24CS60R15_Q1.c, 24CS60R15_Q2.c, 24CS60R15_Q3.c, and your zip file name will be 24CS60R15_A4.zip.
5. **Inputs should be taken from a file and outputs should be printed to a file named output.txt. The input file name will be passed as a command line argument.**
6. You may extend your BST code to AVL tree.

==

Question 1:

Implement the following functionality of the **AVL tree** in a single program with a menu-driven format.

- Insert new node
- Search a node
- Delete a node
- Find_height of the tree
- Print the Preorder of the tree
- Print the Postorder of the tree
- Print the inorder of the tree
- Level order
- zigzag order



Note: try to keep the menu interface the same as the previous assignment, that way you can understand the difference between 2 approaches

You can use the following numbers for each option in the menu:
1 = insert
2 = search
3 = delete
4 = height
5 = Preorder
6 = Postorder
7 = Inorder
8 = Level order
9 = Zigzag order
10 = Indicates the last line in the input

Sample Input

```
1 9
1 45
1 49
1 54
1 77
1 31
1 20
3 9
1 10
2 9
4
5
6
7
8
```

9
10
Sample Output:

Not Found
The height of the tree is 2
Preorder: 45 20 10 31 54 49 77
Postorder: 10 31 20 49 77 54 45
Inorder: 10 20 31 45 49 54 77
Level order: 45 20 54 10 31 49 77
Zig zag order: 45 54 20 10 31 49 77

Question 2:

Construct an AVL tree from the input values and check if every node's value is exactly half of the sum of its child nodes. If this condition holds true for the entire tree, return 1; otherwise, return 0.

Sample Input 1:
1 10
1 15
1 16
1 20
1 24
10

Sample Output 1:
1

Explanation:

```
The tree will be      15
                     /  \
                   10    20
                        /  \
                      16    24

                15 = (10+20)/2
                20 = (16+24)/2
```

Sample Input 2:
1 10
1 15
1 16
1 20
1 25
10

Sample Output 2:
0

Explanation:

```
The tree will be        15
                      /    \
                    10     20
                           /   \
                         16   24

                  15 = (10+20)/2
                  20 != (16+25)/2
```

Question 3:

Given a binary tree. Find the size of its largest subtree which is a Binary Search Tree. You will be provided with the inorder and preorder of the binary tree. In the input file, the first one is the inorder and the second is the preorder.

Sample Input 1:
5
9 30 15 20 27
30 9 20 15 27
Sample Output 2:
The size of the largest BST is 3

Explanation:

```
            30
           /  \
         9    20
              /  \
            15  27
```

This is the largest subtree which is a BST and its size is 3

Sample Input :
3
4 3 2
3 4 2
Sample Output 2:
The size of the largest BST is 1
Explanation:
Only the leaf nodes are the largest BST.