

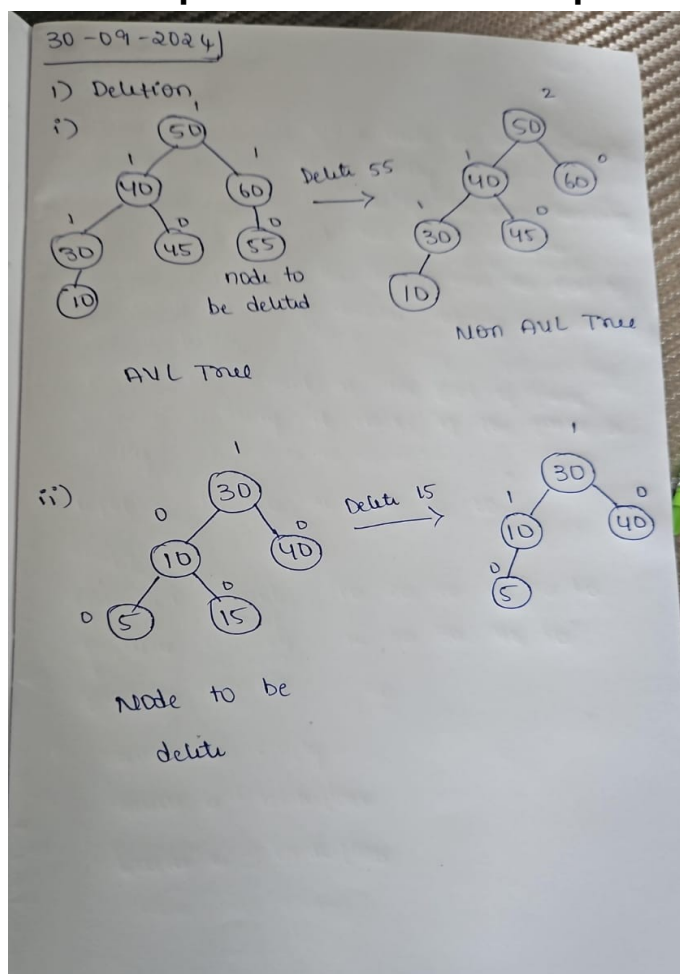
## 1. Write a c program to find the AVL tree

```
main.c
1 #include <stdio.h>
2 #include <stdlib.h>
3 struct Node {
4     int key;
5     struct Node* left;
6     struct Node* right;
7     int height;
8 };
9 int height(struct Node* N) {
10     if (N == NULL)
11         return 0;
12     return N->height;
13 }
14 int max(int a, int b) {
15     return (a > b) ? a : b;
16 }
17 struct Node* createNode(int key) {
18     struct Node* node = (struct Node*)malloc(sizeof(struct Node));
19     node->key = key;
20     node->left = NULL;
21     node->right = NULL;
22     node->height = 1;
23     return node;
24 }
25 struct Node* rightRotate(struct Node* y) {
26     struct Node* x = y->left;
27     struct Node* T2 = x->right;
28     x->right = y;
29     y->left = T2;
30     y->height = max(height(y->left), height(y->right)) + 1;
31     x->height = max(height(x->left), height(x->right)) + 1;
32     return x;
33 }
34 struct Node* leftRotate(struct Node* x) {
35     struct Node* y = x->right;
36     struct Node* T2 = y->left;
37     y->left = x;
38     x->right = T2;
39     x->height = max(height(x->left), height(x->right)) + 1;
40     y->height = max(height(y->left), height(y->right)) + 1;
41     return y;
42 }
43 int main() {
44     struct Node* root = NULL;
45     root = createNode(10);
46     root = createNode(20);
47     root = createNode(30);
48     root = createNode(40);
49     root = createNode(50);
50     printf("In-order traversal of the constructed AVL tree is: 10 20 25 30 40 50\n");
51     return 0;
52 }
```

Output

```
/tmp/3CaY3GzJDr.o
In-order traversal of the constructed AVL tree is: 10 20 25 30 40 50
=== Code Execution Successful ===
```

## 2. Write a problem on deletion operation (AVL Tree)



### 3. Write a problem on searching operation (AVL Tree)

