**Experiment 10**

**AIM:** Write a program to print preorder, inorder, post order traversal of a binary tree.

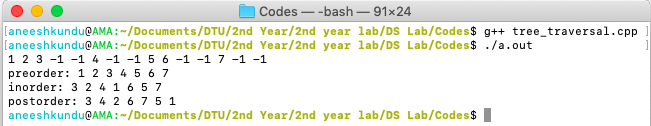
**THEORY:** Unlike linear data structures (Array, Linked List, Queues, Stacks, etc) which have only one logical way to traverse them, trees can be traversed in different ways. Following are the generally used ways for traversing trees.

1. Preorder
2. Inorder
3. Postorder

**Code:**

1. #include < iostream >
2. using namespace std;
3. struct Node {
4. int data;
5. Node \* left, \* right;
6. Node() {
7. left = right = NULL;
8. }
9. Node(int d): data(d), left(NULL), right(NULL) {}
10. };
11. Node \* build\_tree\_preorder() {
12. int d;
13. cin >> d;
14. if (d != -1) {
15. Node \* root = new Node(d);
16. root - > left = build\_tree\_preorder();
17. root - > right = build\_tree\_preorder();
18. return root;
19. } else return NULL;
20. }
21. void preorder(Node \* root) {
22. if (root == NULL) return;
23. cout << root - > data << " ";
24. preorder(root - > left);
25. preorder(root - > right);
26. }
27. void inorder(Node \* root) {
28. if (root == NULL) return;
29. inorder(root - > left);
30. cout << root - > data << " ";
31. inorder(root - > right);
32. }
33. void postorder(Node \* root) {
34. if (root == NULL) return;
35. postorder(root - > left);
36. postorder(root - > right);
37. cout << root - > data << " ";
38. }
39. int main() {
40. Node \* root = build\_tree\_preorder();
41. cout << "preorder: ";
42. preorder(root);
43. cout << endl;
44. cout << "inorder: ";
45. inorder(root);
46. cout << endl;
47. cout << "postorder: ";
48. postorder(root);
49. cout << endl;
50. return 0;
51. }

**Output:**

****