

## Question1.cpp

```
1  #include <iostream>
2  using namespace std;
3  // A binary tree node has data, pointer to left child and a pointer to right child
4  struct Node
5  {
6      int data;
7      struct Node *left, *right;
8  };
9  // Utility function to create a new tree node
10 Node *newNode(int data)
11 {
12     Node *temp = new Node;
13     temp->data = data;
14     temp->left = temp->right = NULL;
15     return temp;
16 }
17
18 void printPreorder(struct Node *node)
19 {
20     if (node == NULL)
21     {
22         return;
23     }
24     cout << node->data << " ";
25     printPreorder(node->left);
26     printPreorder(node->right);
27 }
28
29 int main()
30 {
31     struct Node *root = newNode(1);
32     root->left = newNode(2);
33     root->right = newNode(3);
34     root->left->left = newNode(4);
35     root->left->right = newNode(5);
36
37     // Function call
38     cout << "\npreorder traversal of binary tree is \n";
39     printPreorder(root);
40
41     return 0;
42 }
```

## Question2.cpp

```
1  #include <iostream>
2  using namespace std;
3  // A binary tree node has data, pointer to left child and a pointer to right child
4  struct Node
5  {
6      int data;
7      struct Node *left, *right;
8  };
9  // Utility function to create a new tree node
10 Node *newNode(int data)
11 {
12     Node *temp = new Node;
13     temp->data = data;
14     temp->left = temp->right = NULL;
15     return temp;
16 }
17
18 void printPostorder(struct Node *node)
19 {
20     if (node == NULL)
21     {
22         return;
23     }
24     printPostorder(node->left);
25     printPostorder(node->right);
26     cout << node->data << " ";
27 }
28 int main()
29 {
30     struct Node *root = newNode(1);
31     root->left = newNode(2);
32     root->right = newNode(3);
33     root->left->left = newNode(4);
34     root->left->right = newNode(5);
35
36     // Function call
37     cout << "\nPostorder traversal of binary tree is \n";
38     printPostorder(root);
39     return 0;
40 }
```

## Question3.cpp

```
1  #include <iostream>
2  using namespace std;
3
4  class node
5  {
6  public:
7      int data;
8      node *left;
9      node *right;
10
11     node(int d)
12     {
13         this->data = d;
14         this->left = NULL;
15         this->right = NULL;
16     }
17 };
18 node *buildTree(node *root)
19 {
20     cout << "Enter data" << endl;
21     int data;
22     cin >> data;
23     root = new node(data);
24     if (data == -1)
25     {
26         return NULL;
27     }
28     cout << "Enter data for inserting in left " << data << endl;
29     root->left = buildTree(root->left);
30     cout << "Enter data for inserting in right " << data << endl;
31     root->right = buildTree(root->right);
32 };
33
34 void inorder(node *root)
35 {
36     if (root == NULL)
37     {
38         return;
39     }
40     inorder(root->left);
41     cout << root->data << " ";
42     inorder(root->right);
43 }
44
45 void preorder(node *root)
46 {
47     if (root == NULL)
48     {
49         return;
50     }
51     cout << root->data << " ";
52     preorder(root->left);
53     preorder(root->right);
```

```
54 }
55
56 void postorder(node *root)
57 {
58     if (root == NULL)
59     {
60         return;
61     }
62     postorder(root->left);
63     postorder(root->right);
64     cout << root->data << " ";
65 }
66
67 int main()
68 {
69     node *root = NULL;
70
71     // creating of binary tree
72     root = buildTree(root);
73
74     cout << "inorder traversal is: ";
75     inorder(root);
76
77     cout << endl
78         << "preorder traversal is: ";
79     preorder(root);
80
81     cout << endl
82         << "postorder traversal is: ";
83     postorder(root);
84
85     return 0;
86 }
```