```
 1: /*-----Binary Tree-------*/
 2:
 3: #include<iostream>
 4: #include<queue>
 5: using namespace std;
 6:
 7: class node
 8: {
 9:     public:
10:     int data;
11:     node * left;
12:     node * right;
13:     void info(int p, node * l, node * r)
14:     {
15:         data = p;
16:         left = l;
17:         right = r;
18:     }
19: };
20:
21: class Tree
22: {
23:     public:
24:         node * root;
25:         node a, b, c, d, e, f, g, h, i, j, k;
26:         Tree()
27:         {
28:             root = &f;
29:             a.info(12, &h, &c);
30:             f.info(36, &g, &e);
31:             g.info(42, NULL, &a);
32:             h.info(57, NULL, NULL);
33:             c.info(62, &i, &j);
34:             i.info(74, NULL, NULL);
35:             j.info(24, NULL, NULL);
36:             e.info(17, &b, &k);
37:             b.info(81, NULL, NULL);
38:             k.info(53, &d, NULL);
39:             d.info(69, NULL, NULL);
40:         }
41:         void preOrderTrav(node *x)
42:         {
43:             if(x != NULL)
44:             {
45:                 cout << x -> data << " ";
46:                 preOrderTrav(x -> left);
```

```
47:                 preOrderTrav(x -> right);
48:             }
49:         }
50:
51:         void postOrderTrav(node *x)
52:         {
53:             if(x != NULL)
54:             {
55:                 postOrderTrav(x -> left);
56:                 postOrderTrav(x -> right);
57:                 cout << x -> data << " ";
58:             }
59:         }
60:
61:         void inOrderTrav(node *x)
62:         {
63:             if(x != NULL)
64:             {
65:                 postOrderTrav(x -> left);
66:                 cout << x -> data << " ";
67:                 postOrderTrav(x -> right);
68:             }
69:         }
70:
71:         void BreadthOrderTrav(node *x)
72:         {
73:             queue<node*> temp;
74:             temp.push(x);
75:             while(!temp.empty())
76:             {
77:                 x = temp.front();
78:                 temp.pop();
79:                 cout << x -> data << " ";
80:                 if(x -> left != NULL)
81:                 {
82:                     temp.push(x -> left);
83:                 }
84:                 if(x -> right != NULL)
85:                 {
86:                     temp.push(x -> right);
87:                 }
88:             }
89:         }
90:
91:         /*void DepthOrderTrav(node *x)
92:         {
```

```
 93:             queue<node*> temp;
 94:             temp.push(x);
 95:             while(!temp.empty())
 96:             {
 97:                 x = temp.front();
 98:                 temp.pop();
 99:                 cout << x -> data << " ";
100:                 if(x -> left != NULL)
101:                 {
102:                     temp.push(x -> left);
103:                 }
104:                 if(x -> right != NULL)
105:                 {
106:                     temp.push(x -> right);
107:                 }
108:             }
109:         }*/
110:
111: };
112:
113: int main()
114: {
115:     Tree tree;
116:     tree.preOrderTrav(tree.root);
117:     cout << "\n";
118:     tree.postOrderTrav(tree.root);
119:     cout << "\n";
120:     tree.inOrderTrav(tree.root);
121:     cout << "\n";
122:
123:     tree.BreadthOrderTrav(tree.root);   // Same as preOrder?
124:     cout << "\n";
125:
126:     //tree.DepthOrderTrav(tree.root);
127:
128:     return 0;
129: }
```