

```

1: #include<stdio.h>
2: #include<stdlib.h>
3: struct node
4: {
5:     int data;
6:     struct node* left;
7:     struct node* right;
8: };
9: struct node* root;
10: struct node* insert(struct node* r,int datatonode)
11: {
12:     if(r == NULL)
13:     {
14:         r = (struct node*)malloc (sizeof(struct node));
15:         r->data = datatonode;
16:         r->left = NULL;
17:         r->right = NULL;
18:     }
19:     else if(datatonode < r->data)
20:         r->left= insert(r->left,datatonode);
21:     else
22:         r->right = insert(r->right,datatonode);
23:     return r;
24: }
25: void inOrder(struct node* r)
26: {
27:     if(r!= NULL)
28:     {
29:         inOrder(r->left);
30:         printf("%d",r->data);
31:         inOrder(r->right);
32:     }
33: }
34: void preOrder(struct node* r)
35: {
36:     if(r!= NULL)
37:     {
38:         printf("%d",r->data);
39:         preOrder(r->left);
40:         preOrder(r->right);
41:     }
42: }
43: void postOrder(struct node* r)
44: {
45:     if(r!= NULL)
46:     {
47:         postOrder(r->left);
48:         postOrder(r->right);
49:         printf("%d",r->data);
50:     }
51: }
52: int main()
53: {
54:     root = NULL;
55:     int number,value;
56:     printf("Enter the number of elements to be inserted?\n");
57:     scanf("%d",&number);
58:     for(int i=0;i<number;i++)
59:     {
60:         printf("Data no %d is ",i+1);
61:         scanf("%d",&value);

```

```
62: root = insert(root,value);
63: }
64: printf("INRODER TARVERSAL\n");
65: inOrder(root);
66: printf("\n");
67: printf("PREORDER TRAVERSAL \n");
68: preOrder(root);
69: printf("\n");
70: printf("POSTORDRER TRAVERSAL \n");
71: postOrder(root);
72: printf("\n");
73: return 0;
74: }
```