

```

1  #include<stdio.h>
2  #include<conio.h>
3  #include<stdlib.h>
4  struct node
5  {
6      int info;
7      struct node *rlink;
8      struct node *llink;
9  };
10 typedef struct node *NODE;
11 NODE getnode()
12 {
13     NODE x;
14     x=(NODE)malloc(sizeof(struct node));
15     if(x==NULL)
16     {
17         printf("mem full\n");
18         exit(0);
19     }
20     return x;
21 }
22 void freenode(NODE x)
23 {
24     free(x);
25 }
26 NODE insert(NODE root,int item)
27 {
28     NODE temp,cur,prev;
29     temp=getnode();
30     temp->rlink=NULL;
31     temp->llink=NULL;

```

```

30     temp->rlink=NULL;
31     temp->llink=NULL;
32     temp->info=item;
33     if(root==NULL)
34         return temp;
35     prev=NULL;
36     cur=root;
37     while(cur!=NULL)
38     {
39         prev=cur;
40         cur=(item<cur->info)?cur->llink:cur->rlink;
41     }
42     if(item<prev->info)
43         prev->llink=temp;
44     else
45         prev->rlink=temp;
46     return root;
47 }
48 void display(NODE root,int i)
49 {
50     int j;
51     if(root!=NULL)
52     {
53         display(root->rlink,i+1);
54         for(j=0;j<i;j++)
55             printf(" ");
56         printf("%d\n",root->info);
57         display(root->llink,i+1);
58     }
59 }
60 NODE delete(NODE root,int item)
61 {

```

```

60 NODE delete(NODE root,int item)
61 {
62     NODE cur,parent,q,suc;
63     if(root==NULL)
64     {
65         printf("empty\n");
66         return root;
67     }
68     parent=NULL;
69     cur=root;
70     while(cur!=NULL&&item!=cur->info)
71     {
72         parent=cur;
73         cur=(item<cur->info)?cur->llink:cur->rlink;
74     }
75     if(cur==NULL)
76     {
77         printf("not found\n");
78         return root;
79     }
80     if(cur->llink==NULL)
81         q=cur->rlink;
82     else if(cur->rlink==NULL)
83         q=cur->llink;
84     else
85     {
86         suc=cur->rlink;
87         while(suc->llink!=NULL)
88             suc=suc->llink;
89         suc->llink=cur->llink;
90         q=cur->rlink;
91     }

```

```

92     }
93     if(parent==NULL)
94         return q;
95     if(cur==parent->llink)
96         parent->llink=q;
97     else
98         parent->rlink=q;
99     freenode(cur);
100     return root;
101 }
102 void preorder(NODE root)
103 {
104     if(root!=NULL)
105     {
106         printf("%d\n",root->info);
107         preorder(root->llink);
108         preorder(root->rlink);
109     }
110 }
111 void postorder(NODE root)
112 {
113     if(root!=NULL)
114     {
115         postorder(root->llink);
116         postorder(root->rlink);
117         printf("%d\n",root->info);
118     }
119 }
120 }
121 void inorder(NODE root)

```

```

120 }
121 void inorder(NODE root)
122 {
123     if(root!=NULL)
124     {
125
126         inorder(root->llink);
127         printf("%d\n",root->info);
128         inorder(root->rlink);
129     }
130 }
131 void main()
132 {
133     int item,choice;
134     NODE root=NULL;
135     for(;;)
136     {
137         printf("\n1.insert\n2.display\n3.pre\n4.post\n5.in\n6.delete\n7.exit\n");
138         printf("enter the choice\n");
139         scanf("%d",&choice);
140         switch(choice)
141         {
142             case 1:printf("enter the item\n");
143                     scanf("%d",&item);
144                     root=insert(root,item);
145                     break;
146             case 2:display(root,0);
147                     break;
148             case 3:preorder(root);
149                     break;
150             case 4:postorder(root);
151                     break;

```

```

152             case 5:inorder(root);
153                     break;
154             case 6:printf("enter the item\n");
155                     scanf("%d",&item);
156                     root=delete(root,item);
157                     break;
158             default:exit(0);
159                     break;
160         }
161     }
162 }
163

```

```

1.insert
2.display
3.pre
4.post
5.in
6.delete
7.exit
enter the choice
1
enter the item
50

1.insert
2.display
3.pre
4.post
5.in
6.delete
7.exit
enter the choice
1
enter the item
20

1.insert
2.display
3.pre
4.post
5.in
6.delete

```

```
6.delete
7.exit
enter the choice
1
enter the item
40

1.insert
2.display
3.pre
4.post
5.in
6.delete
7.exit
enter the choice
1
enter the item
80

1.insert
2.display
3.pre
4.post
5.in
6.delete
7.exit
enter the choice
1
enter the item
90
```

```
1.insert
2.display
3.pre
4.post
5.in
6.delete
7.exit
enter the choice
1
enter the item
15

1.insert
2.display
3.pre
4.post
5.in
6.delete
7.exit
enter the choice
1
enter the item
70

1.insert
2.display
3.pre
4.post
5.in
6.delete
7.exit
```

```
7.exit
enter the choice
1
enter the item
70

1.insert
2.display
3.pre
4.post
5.in
6.delete
7.exit
enter the choice
2
    90
    80
    70
50
    40
    20
    15

1.insert
2.display
3.pre
4.post
5.in
6.delete
7.exit
enter the choice
```

```
7.exit
enter the choice
3
50
20
15
40
80
70
90

1.insert
2.display
3.pre
4.post
5.in
6.delete
7.exit
enter the choice
5
15
20
40
50
70
80
90

1.insert
2.display
3.pre
```

```
1.insert
2.display
3.pre
4.post
5.in
6.delete
7.exit
enter the choice
```

```
5
15
20
40
50
70
80
90
```

```
1.insert
2.display
3.pre
4.post
5.in
6.delete
7.exit
enter the choice
7
```

```
...Program finished with exit code 0
Press ENTER to exit console.
```