



treebinary.cxx



/storage/emulated...

```
1 #include<stdlib.h>
2 #include<stdio.h>
3 struct tree {
4     int info;
5     struct tree *left;
6     struct tree *right;
7 };
8 struct tree *insert(struct tree *,int);
9 void inorder(struct tree *);
10 void postorder(struct tree *);
11 void preorder(struct tree *);
12 struct tree *deletei(struct tree *,int);
13 struct tree *search(struct tree *);
14 int main(void){
15     struct tree *root;
16     int choice,item,item_no;
17     root=NULL;
18     /*rear=NULL;*/
19     do{
20         do{
21             printf("\n \t 1.Insert in binary tree");
22             printf("\n \t 2.Delete from binary tree");
23             printf("\n \t 3.Inorder traversal of binary
tree");
24             printf("\n \t 4.Search");
25             printf("\n \t 5.Exit");
26             printf("\n \t Enter choice:");
27             scanf("%d",&choice);
```



treebinary.cxx



/storage/emulated...

```
27     scanf("%d",&choice);
28     if(choice<1 || choice>7)
29         printf("\n Invalid choice - try again");
30 }
31 while(choice<1 || choice>7);
32 switch(choice){
33     case 1:
34         printf("\n Enter new element:");
35         scanf("%d",&item);
36         root=insert(root,item);
37         printf("\n root is %d",root->info);
38         printf("\n Inorder traversal of binary
tree is :");
39         inorder(root);
40         break;
41     case 2:
42         printf("\n Enter the element to be
deleted:");
43         scanf("%d",&item_no);
44         root=deletei(root,item_no);
45         inorder(root);
46         break;
47     case 3:
48         printf("\n Inorder traversal of
binary tree is:");
49         inorder(root);
50         break;
51     case 4:
```



treebinary.cxx



/storage/emulated...

```
52         printf("\n Search operation in
binary tree");
53         root=search(root);
54         break;
55         default:
56             printf("\n End of program");
57     }
58 }
59 while(choice !=5);
60 return(0);
61 }
62 struct tree *insert(struct tree *root,int x){
63     if(!root){
64         root=(struct tree*
        )malloc(sizeof(struct tree));
65         root->info=x;
66         root->left=NULL;
67         root->right=NULL;
68         return(root);
69     }
70     if(root->info>x)
71         root->left=insert(root->left,x);else {
72         if(root->info<x)
73             root->right=insert(root->right,x);
74     }
75     return(root);
76 }
77 void inorder(struct tree *root) {
```



treebinary.cxx



/storage/emulated...

```
77 void inorder(struct tree *root) {
78     if(root!=NULL) {
79         inorder(root->left);
80         printf("%d",root->info);
81         inorder(root->right);
82     }
83     return;
84 }
85 struct tree *deletei(struct tree *ptr,int x) {
86     struct tree *p1, *p2;
87     if(!ptr){
88         printf("\n Node not found");
89         return(ptr);
90     }else{
91         if(ptr->info<x){
92             ptr->right=deletei(ptr->right,x);
93             /*return(ptr);*/
94         }else if(ptr->info>x) {
95             ptr->left=deletei(ptr->left,x);
96             return ptr;
97         }else
98         {
99             if(ptr->info==x)
100             {
101                 if(ptr->left==ptr->right)
102                 {
103                     free(ptr);
104                     return(NULL);
```



treebinary.cxx



/storage/emulated...

```
105         }else if(ptr->left==NULL)
106         {
107             p1=ptr->right;
108             free(ptr);
109             return p1;
110         }else if(ptr->right==NULL)
111         {
112             p1=ptr->left;
113             free(ptr);
114             return p1;
115         }else {
116             p1=ptr->right;
117             p2=ptr->right;
118             while(p1->left!=NULL)
119                 p1=p1->left;
120             p1->left=ptr->left;
121             free(ptr);
122             return p2;
123         }
124     }
125 }
126 }
127 return(ptr);
128 }
129 struct tree *search(struct tree *root) {
130     int no,i,ino;
131     struct tree *ptr;
132     ptr=root;
```



treebinary.cxx



/storage/emulated...

```
132         ptr=root;
133         printf("\n Enter the element to be
searched:");
134         scanf("%d",&no);
135         fflush(stdin);
136         while(ptr) {
137             if(no>ptr->info)
138                 ptr=ptr->right;
139             else if(no<ptr->info)
140                 ptr=ptr->left; else
141                 break;
142         }
143         if(ptr) {
144             printf("\n Element %d which was
searched is found and is =%d",no,ptr->info);
145         }else
146             printf("\n Element %d does not exist in
the binary tree",no);
147         return(ptr);
148     }
149
150
151
152
```







TAB



```
1.Insert in binary tree
2.Delete from binary tree
3.Inorder traversal of binary tree
4.Search
5.Exit
Enter choice:1
```

Enter new element:65

```
root is 65
Inorder traversal of binary tree is :65
1.Insert in binary tree
2.Delete from binary tree
3.Inorder traversal of binary tree
4.Search
5.Exit
Enter choice:1
```

Enter new element:55

```
root is 65
Inorder traversal of binary tree is :5565
1.Insert in binary tree
2.Delete from binary tree
3.Inorder traversal of binary tree
4.Search
5.Exit
Enter choice:1
```

Enter new element:46

```
root is 65
Inorder traversal of binary tree is :465565
1.Insert in binary tree
2.Delete from binary tree
3.Inorder traversal of binary tree
4.Search
5.Exit
Enter choice:1
```

Enter new element:48

```
root is 65
Inorder traversal of binary tree is :46485565
1.Insert in binary tree
```



TAB



```
2.Delete from binary tree
3.Inorder traversal of binary tree
4.Search
5.Exit
Enter choice:1
```

Enter new element:46

```
root is 65
Inorder traversal of binary tree is :465565
1.Insert in binary tree
2.Delete from binary tree
3.Inorder traversal of binary tree
4.Search
5.Exit
Enter choice:1
```

Enter new element:48

```
root is 65
Inorder traversal of binary tree is :46485565
1.Insert in binary tree
2.Delete from binary tree
3.Inorder traversal of binary tree
4.Search
5.Exit
Enter choice:3
```

```
Inorder traversal of binary tree is:46485565
1.Insert in binary tree
2.Delete from binary tree
3.Inorder traversal of binary tree
4.Search
5.Exit
Enter choice:4
```

Search operation in binary tree  
Enter the element to be searched:48

```
Element 48 which was searched is found and is =48
1.Insert in binary tree
2.Delete from binary tree
3.Inorder traversal of binary tree
4.Search
5.Exit
Enter choice:█
```