



*Untitled - Notepad

File Edit Format View Help

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
struct node
{
int info;
struct node*llink;
struct node*rlink;
};
typedef struct node*NODE;
NODE getnode()
{
NODE x;
x=(NODE)malloc(sizeof(struct node));
if(x==NULL)
{
printf("Memory not available!");
exit(0);
}
return x;
}
void freenode(NODE x)
{
free(x);
}
NODE insert(int item,NODE root)
{
NODE temp,cur,prev;
char direction[10];
int i;
temp=getnode();
temp->info=item;
temp->llink=NULL;
temp->rlink=NULL;
if(root==NULL)
return temp;
printf("Give direction to insert..\n");
scanf("%s",direction);
prev=NULL;
cur=root;
for(i=0;i<strlen(direction)&&cur!=NULL;i++)
{
prev=cur;
if(direction[i]=='l')
cur=cur->llink;
else
cur=cur->rlink;
}
if(cur!=NULL||i!=strlen(direction))
{
```

*Untitled - Notepad

File Edit Format View Help

```
}
```

```
if(cur!=NULL||i!=strlen(direction))
```

```
{
```

```
printf("Insertion not possible\n");
```

```
freenode(temp);
```

```
return(root);
```

```
}
```

```
if(cur==NULL)
```

```
{
```

```
if(direction[i-1]=='l')
```

```
prev->llink=temp;
```

```
else
```

```
prev->rlink=temp;
```

```
}
```

```
return(root);
```

```
}
```

```
void preorder(NODE root)
```

```
{
```

```
if(root!=NULL)
```

```
{
```

```
printf("the item is %d\n",root->info);
```

```
preorder(root->llink);
```

```
preorder(root->rlink);
```

```
}
```

```
}
```

```
void inorder(NODE root)
```

```
{
```

```
if(root!=NULL)
```

```
{
```

```
inorder(root->llink);
```

```
printf("The item is %d\n",root->info);
```

```
inorder(root->rlink);
```

```
}
```

```
}
```

```
void postorder(NODE root)
```

```
{
```

```
if (root!=NULL)
```

```
{
```

```
postorder(root->llink);
```

```
postorder(root->rlink);
```

```
printf("The item is %d\n",root->info);
```

```
}
```

```
}
```

```
void display(NODE root,int i)
```

```
{
```

```
int j;
```

```
if(root!=NULL)
```

```
{
```

```
display(root->rlink,i+1);
```

```
for(j=1;j<i;j++)
```

*Untitled - Notepad

```
File Edit Format View Help
display(root->rlink,i+1);
for (j=1;j<=i;j++)
printf(" ");
printf("%d\n",root->info);
display(root->llink,i+1);
}
}

int main()
{
NODE root=NULL;
int choice,i,item;

for(;;)
{
printf("1.Insert\n2.Preorder\n3.Inorder\n4.Postorder\n5.Display\n");
printf("Enter the choice:\n");
scanf("%d",&choice);
switch(choice)
{
case 1: printf("Enter the item:\n");
scanf("%d",&item);
root=insert(item,root);
break;
case 2: if(root==NULL)
{
printf("Tree is empty!");
}
else
{
printf("Given tree is..");
display(root,1);
printf("The preorder traversal is:\n");
preorder(root);
}
break;
case 3:if(root==NULL)
{
printf("Tree is empty");
}
else
{
printf("Given tree is..");
display(root,1);
printf("The inorder traversal is \n");
inorder(root);
}
break;
case 4:if (root==NULL)
{
```

*Untitled - Notepad

File Edit Format View Help

```
{  
case 1: printf("Enter the item:\n");  
        scanf("%d",&item);  
        root=insert(item,root);  
        break;  
case 2: if(root==NULL)  
        {  
            printf("Tree is empty!");  
        }  
        else  
        {  
            printf("Given tree is..");  
            display(root,1);  
            printf("The preorder traversal is:\n");  
            preorder(root);  
        }  
        break;  
case 3:if(root==NULL)  
        {  
            printf("Tree is empty");  
        }  
        else  
        {  
            printf("Given tree is..");  
            display(root,1);  
            printf("The inorder traversal is \n");  
            inorder(root);  
        }  
        break;  
case 4:if (root==NULL)  
        {  
            printf("Tree is empty");  
        }  
        else  
        {  
            printf("Given tree is..");  
            display(root,1);  
            printf("The postorder traversal is \n");  
            postorder(root);  
        }  
        break;  
case 5:display(root,1);  
        break;  
default:printf("Invalid choice entered.\n");  
        exit(0);  
}  
}  
return 0;  
}
```

```
> clang-7 -pthread -lm -o main main.c
```

```
> ./main
```

- 1.Insert
- 2.Preorder
- 3.Inorder
- 4.Postorder
- 5.Display

```
Enter the choice:
```

```
1
```

```
Enter the item:
```

```
23
```

- 1.Insert
- 2.Preorder
- 3.Inorder
- 4.Postorder
- 5.Display

```
Enter the choice:
```

```
1
```

```
Enter the item:
```

```
34
```

```
Give direction to insert..
```

```
l
```

- 1.Insert
- 2.Preorder
- 3.Inorder
- 4.Postorder
- 5.Display

```
Enter the choice:
```

```
1
```

```
Enter the item:
```

```
56
```

```
Give direction to insert..
```

```
r
```

r

- 1.Insert
- 2.Preorder
- 3.Inorder
- 4.Postorder
- 5.Display

Enter the choice:

1

Enter the item:

78

Give direction to insert..

ll

- 1.Insert
- 2.Preorder
- 3.Inorder
- 4.Postorder
- 5.Display

Enter the choice:

1

Enter the item:

58

Give direction to insert..

lr

- 1.Insert
- 2.Preorder
- 3.Inorder
- 4.Postorder
- 5.Display

Enter the choice:

1

Enter the item:

96

Give direction to insert..

Give direction to insert..

- rl
- 1.Insert
- 2.Preorder
- 3.Inorder
- 4.Postorder
- 5.Display

Enter the choice:

1

Enter the item:

85

Give direction to insert..

- rr
- 1.Insert
- 2.Preorder
- 3.Inorder
- 4.Postorder
- 5.Display

Enter the choice:

5

85

56

96

23

58

34

78

- 1.Insert
- 2.Preorder
- 3.Inorder
- 4.Postorder
- 5.Display

Enter the choice:

Enter the choice:

2

Given tree is.. 85

56

96

23

58

34

78

The preorder traversal is:

the item is 23

the item is 34

the item is 78

the item is 58

the item is 56

the item is 96

the item is 85

1.Insert

2.Preorder

3.Inorder

4.Postorder

5.Display

Enter the choice:

3

Given tree is.. 85

56

96

23

58

34

78

The inorder traversal is

The item is 78

The item is34
The item is58
The item is23
The item is96
The item is56
The item is85

- 1.Insert
- 2.Preorder
- 3.Inorder
- 4.Postorder
- 5.Display

Enter the choice:

4

Given tree is.. 85

```
    56
      96
    23
      58
    34
      78
```

The postorder traversal is

The item is78
The item is58
The item is34
The item is96
The item is85
The item is56
The item is23

- 1.Insert
- 2.Preorder
- 3.Inorder
- 4.Postorder
- 5.Display

TREES

LAB PROGRAM - 10

BINARY SEARCH

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct node
{
    int info;
    struct node *link;
};

typedef struct node *NODE;
NODE getnode()
{
    NODE x;
    if (x == NULL)
        exit(0);
    return x;
}

void freeNode(NODE x)
{
    free(x);
}

NODE insert(int item, NODE root)
{
    NODE temp, cur, prev;
    char direction[10];
    int i;
    temp = getnode();
    temp->info = item;
    temp->link = NULL;
    if (root == NULL)
        root = temp;
    else
    {
        cur = root;
        while (cur->link != NULL)
            cur = cur->link;
        cur->link = temp;
    }
    if (item < root->info)
        direction[0] = 'L';
    else
        direction[0] = 'R';
    for (i = 1; i < 10; i++)
        direction[i] = '\0';
    printf("Enter direction to insert... \n");
    scanf("%s", direction);
    if (direction[0] == 'L')
        root = rotateLeft(root);
    else
        root = rotateRight(root);
}

```

```

curr = root;
for (i=0; i<strlen(direction) && curr != NULL; i++)
    if (curr == curr)
        if (direction[i] == 'l')
            curr = curr->lchild;
        else
            curr = curr->rchild;
    if (curr != NULL || i == strlen(direction))
        priority C("Insersion not possible\n");
        prenode(stmp);
        return root;
    if (curr == NULL)
        if (direction[i-1] == 'l')
            curr->lchild = stmp;
        else
            curr->rchild = stmp;
        stmp->parent = curr;
        return curr;
    void preorder (NODE root)
    if (root != NULL)
        priority C("the item is %d\n", root->info);
        preorder (root->lchild);
        preorder (root->rchild);
        printf ("root %d\n", root->info);
    void inorder (NODE root)
    if (root != NULL)
        if (inorder (root->lchild));
            priority C("the item is %d\n", root->info);
            inorder (root->rchild);
        postorder (root->rchild);
        postorder (root->lchild);
        priority C("The item is %d\n", root->info);
    void display (NODE root, int i)
    
```

{ init g;

if (root != NULL)

{ display (root → rlink, i+1);

for (j=1; j <= i; j++)

printf (" ");

printf ("%d\n", root → rlink);

display (root → llink, i+1); }

int main()

{

NODE root = NULL;

int choice, i, item;

char l;

1. Insert 2. Preorder 3. Inorder 4. Postorder
5. Display;

priority C ("Enter the choice : ln");

Scnry C ("%d", &choice);

Switch (choice);

Case 1: priority C ("Enter the item : ln");

Scnry C ("%d", &item);

root = insert (item, root);

break;

Case 2: if (root == NULL)

{ printf ("Tree is empty!"); }

else

{ priority C ("Given tree is ..");

display (root, 1);

printf ("The preorder traversal is : \n");

preorder (root); }

break;

Case 3: if (root == NULL)

{ priority C ("Tree is empty!"); }

else

{ print ("Given tree is .. ");

display (root, 1);

print ("The postorder traversal is ");

inorder (root); };

break;

Case 4: if (root == NULL)

{ print ("Tree is empty "); };

else

{ print ("Given tree is .. ");

display (root, 1);

print ("The postorder traversal is ");

postorder (root); };

break;

Case 5: display (root, 1);

break;

default: print ("Invalid choice entered ");

exit (0);

};

};

return 0;

};