DS LAB-PROG-Extra Progs

Binary Tree and BST with extra functions

Program and Output

Mallika Prasad

1BM19CS081

30.12.2020

Program1-

Binary Tree

```
#include<stdio.h>
#include<stdlib.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))
{
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++)
printf("  ");
printf("%d\n",root->info);
display(root->llink,i+1);
}
}

int main()
{
NODE root=NULL;
int choice,i,item;
//clrscr();
do
{
printf("1.insert\n2.preorder\n3.inorder\n4.postorder\n5.display\n6.exit");
printf("\nenter 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)
                {
                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;
            case 6:exit(0);
default:exit(0);
}
}while(choice!=6);
}
```

Output-

```
1.insert
2.preorder
3.inorder
4.postorder
5.display
6.exit
enter the choice
1
enter the item
1
1.insert
2.preorder
3.inorder
4.postorder
5.display
6.exit
enter the choice
1
enter the item
2
give direction to insert
l
1.insert
2.preorder
3.inorder
4.postorder
5.display
6.exit
enter the choice
1
enter the item
```

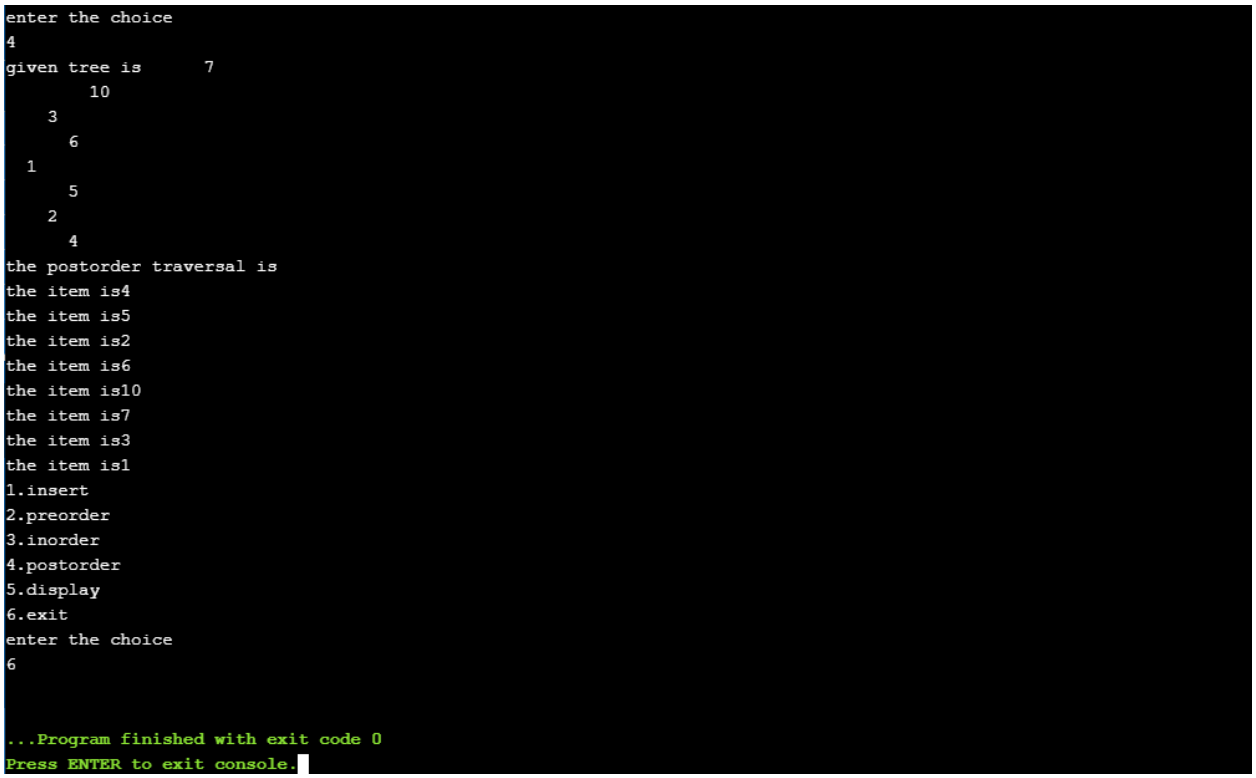
```
enter the item
3
give direction to insert
r
1.insert
2.preorder
3.inorder
4.postorder
5.display
6.exit
enter the choice
1
enter the item
4
give direction to insert
ll
1.insert
2.preorder
3.inorder
4.postorder
5.display
6.exit
enter the choice
1
enter the item
5
give direction to insert
lr
1.insert
2.preorder
3.inorder
```

```
1.insert
2.preorder
3.inorder
4.postorder
5.display
6.exit
enter the choice
1
enter the item
6
give direction to insert
rl
1.insert
2.preorder
3.inorder
4.postorder
5.display
6.exit
enter the choice
1
enter the item
7
give direction to insert
rr
1.insert
2.preorder
3.inorder
4.postorder
5.display
6.exit
enter the choice
```

```
6.exit
enter the choice
1
enter the item
10
give direction to insert
rrl
1.insert
2.preorder
3.inorder
4.postorder
5.display
6.exit
enter the choice
5
      7
        10
    3
      6
  1
      5
    2
      4
1.insert
2.preorder
3.inorder
4.postorder
5.display
6.exit
enter the choice
2
```

```
2
given tree is      7
        10
    3
      6
  1
      5
    2
      4
the preorder traversal is
the item is 1
the item is 2
the item is 4
the item is 5
the item is 3
the item is 6
the item is 7
the item is 10
1.insert
2.preorder
3.inorder
4.postorder
5.display
6.exit
enter the choice
3
given tree is      7
        10
    3
      6
  1
```

```
given tree is      7
        10
    3
      6
  1
      5
    2
      4
the inorder traversal is
the item is4
the item is2
the item is5
the item is1
the item is6
the item is3
the item is10
the item is7
1.insert
2.preorder
3.inorder
4.postorder
5.display
6.exit
enter the choice
4
given tree is      7
        10
    3
      6
  1
      5
```



Program 2-

1]Count the number of nodes in BST

2]Find the maximum and minimum element in BST

```
#include<stdio.h>

#include<stdlib.h>

struct node

 {

  int info;

  struct node *rlink;

  struct node *llink;

 };
```

```
typedef struct node *NODE;
NODE getnode()
{
NODE x;
x=(NODE)malloc(sizeof(struct node));
if(x==NULL)
 {
  printf("mem full\n");
  exit(0);
 }
 return x;
}
void freenode(NODE x)
{
free(x);
}
NODE insert(NODE root,int item)
{
NODE temp,cur,prev;
temp=getnode();
temp->rlink=NULL;
temp->llink=NULL;
temp->info=item;
if(root==NULL)
```

```
 return temp;
prev=NULL;
cur=root;
while(cur!=NULL)
{
prev=cur;
cur=(item<cur->info)?cur->llink:cur->rlink;
}
if(item<prev->info)
 prev->llink=temp;
else
 prev->rlink=temp;
return root;
}
void display(NODE root,int i)
{
int j;
if(root!=NULL)
 {
  display(root->rlink,i+1);
  for(j=0;j<i;j++)
              printf("  ");
   printf("%d\n",root->info);
             display(root->llink,i+1);
```

```
 }
}

void preorder(NODE root)
{
if(root!=NULL)
 {
  printf("%d\n",root->info);
  preorder(root->llink);
  preorder(root->rlink);
  }
 }
void postorder(NODE root)
{
if(root!=NULL)
 {

  postorder(root->llink);
  postorder(root->rlink);
  printf("%d\n",root->info);
  }
 }
void inorder(NODE root)
{
```

```
if(root!=NULL)
 {

  inorder(root->llink);
  printf("%d\n",root->info);
  inorder(root->rlink);
  }
 }
int count(NODE root)
{
   int c=1;
   if (root ==NULL)
      return 0;

   else
   {
      c += count(root->llink);
      c += count(root->rlink);
      return c;
   }
}

void largest(NODE root)
{
```

```
   while (root != NULL && root->rlink != NULL)
   {
      root = root->rlink;
   }
   printf("Largest value is %d\n", root->info);
}
void smallest(NODE root)
{
   while (root != NULL && root->llink != NULL)
   {
      root = root->llink;
   }
   printf("Smallest value is %d\n", root->info);
}

int main()
{
int item,choice;
NODE root=NULL;
printf("\n1.insert\n2.display\n3.preorder\n4.postorder\n5.inorder\n6.count number of
nodes\n7.largest element\n8.smallest element\n9.exit\n");
do
{
printf("enter the choice\n");
```

```
scanf("%d",&choice);
switch(choice)
 {
  case 1:printf("enter the item\n");
                    scanf("%d",&item);
                    root=insert(root,item);
                    break;
  case 2:display(root,0);
                    break;
  case 3:preorder(root);
                    break;
  case 4:postorder(root);
                    break;
  case 5:inorder(root);
                    break;
  case 6:
            printf("Number of nodes: %d\n",count(root));
            break;
  case 7:largest(root);
                    break;
  case 8:smallest(root);
       break;
  case 9: break;
  default:exit(0);
```

```
                break;
            }
        }while(choice!=9);
        return 0;
}
```

Output-

```
1
enter the item
90
enter the choice
1
enter the item
10
enter the choice
1
enter the item
40
enter the choice
1
enter the item
100
enter the choice
2
      100
    90
  70
    60
50
    40
  20
    10
enter the choice
6
Number of nodes: 8
enter the choice
7
Largest value is 100
```

```
enter the choice
6
Number of nodes: 8
enter the choice
7
Largest value is 100
enter the choice
8
Smallest value is 10
enter the choice
3
50
20
10
40
70
60
90
100
enter the choice
4
10
40
20
60
100
90
70
50
enter the choice
5
```

```
40
70
60
90
100
enter the choice
4
10
40
20
60
100
90
70
50
enter the choice
5
10
20
40
50
60
70
90
100
enter the choice
9


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