Lab-10

Dinyashree.k
IBM19CS054

```
#include <stdio.h>
#include <stdlib.h>
struct tree
{
  int value;
  struct tree *l;
  struct tree *r;
}*root = NULL, *temp = NULL, *t2, *t1;
void insert();
void inorder(struct tree *t);
void search(struct tree *t);
void preorder(struct tree *t);
void postorder(struct tree *t);
void search1(struct tree *t, int data);
int flag = 1;
int main()
{
  int ch;
  printf("1. Insert an element into tree\n");
  printf("2. Inorder Traversal\n");
  printf("3 Preorder Traversal\n");
  printf("4. Postorder Traversal\n");
  printf("5. Exit\n");
  do
  {
  printf("\n Enter your choice:");
  scanf("%d", &ch);
  switch(ch)
  {
  case 1:
  insert();
  break;
  case 2:
  inorder(root);
  break;
```

```
case 3:
preorder(root);
break;
case 4:
postorder(root);
break;
case 5:
exit(0);
break;
}while(ch!=5);
return 0;
}
void insert()
{
int data
printf("Enter data of node to be inserted :");
scanf("%d",&data);
temp = (struct tree *)malloc(1*sizeof(struct tree));
temp->value=data;
temp->l=temp->r=NULL
if(root==NULL)
root=temp
else
search(root);
}
void search(struct tree*t)
{
if((temp->value>t->value)&&(t->r!=NULL))
search(t->r);
else if((temp->value>t->value)&&
(t->r==NULL))
t->r=temp;
else if((temp->value<t->value)&&
(t->l!=NULL))
search(t->l);
```

```
else if ((temp->value < t->value) &&
                    (t->l == NULL))
            t->l = temp;
}
void search (struct tree *t, int data);
{
    if ((data > t->value))
    {
        t1 = t;
        search1 (t->r, data);
    } else if ((data < t->value)) {
        t1 = t;
        search1 (t->l, data);
    }
}
void inorder (struct tree *t)
{
    if (root == NULL)
    {
        printf ("Tree empty \n");
        return;
    }
    if (t->l != NULL)
        inorder (t->l);
        printf ("%d -> ", t->value);
        if (t->r != NULL)
            inorder (t->r);
}
void preorder (struct tree *t)
{
    if (root == NULL)
    {
        printf ("Tree empty");
        return;
    }
```

```
printf("%d -> ", t->value);
  if (t->l != NULL)
   preorder(t->l)
  if (t->r != NULL)
    preorder(t->r);
}
void postorder(struct tree *t)
{
   if (root == NULL)
   {
     printf("Tree empty");
     return;
   }
  if (t->l != NULL)
    postorder(t->l)
   if (t->r != NULL)
    postorder(t->r);
    printf("%d ->", t->value);
}
```

Dinesh kumar.K