Name: S Skanda Date: 21/12/20
USN: 1BM19CS137

```
#include<stdio.h>
#include<stdlib.h>
#include<process.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("Memory full \n");
     exit(0);
   }
   return 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+2);
  }
}

NODE delete (NODE root, int item)
{ NODE cur, parent, q, suc;
  if (root == NULL)
  { printf ("tree is empty \n");
    return root;
  }
  parent = NULL;
  cur = root;
  while (cur != NULL && item != cur->info)
  { parent = cur;
    cur = (item < cur->info)? cur->llink : cur->rlink;
  }
```

```
if (cur == NULL)
{ printf ("not found \n");
  return root;
}
if (cur -> llink == NULL)
   q = cur -> rlink;
else if (cur -> rlink = NULL)
   q = cur -> llink;
else
{ suc = cur -> rlink;
  while (suc -> llink != NULL)
     suc = suc -> llink;
  suc -> llink = cur -> llink;
  q = cur -> rlink;
}
if (parent == NULL)
   return q;
if (cur == parent -> llink)
   parent -> llink = q;
else
   parent -> rlink = q;
free (cur);
return root;
}
void preorder (NODE root)
{ if (root != NULL)
  { printf ("%d ", root -> info);
    preorder (root -> llink);
    preorder (root -> rlink);
  }
}
```

```
void posterder (NODE root)
{  if (root!=NULL)
    {  posterder (root->llink);
       posterder (root->rlink);
       printf ("%d ", root->info);
    }
}

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

void main()
{ int item, choice;
  NODE root = NULL;
  for (;;)
  {  printf("\n1.insert 2.display 3.preorder 4.post
            order 5.inorder 6.delete 7.exit \n");
     printf ("%d", &choice);
     switch (choice)
     {  case 1: printf ("enter the item : ");
                scanf ("%d ", &item);
                root = insert (root, item);
                break;
        case 2: if (root != NULL)
                   display (root);
                else
                   printf ("tree is empty");
                break;
```

```
case 3: if (root != NULL)
            preorder (root);
        else
            printf("tree is empty \n");
        break
case 4: if (root != NULL)
            postorder (root);
        else
            printf("tree is empty \n");
case 5: if (root != NULL)
            inorder (root);
        else
            printf ("tree is empty \n");
case 6: printf ("enter the item: ");
        scanf ("%d", &item);
        root = delete (root, item);
        break;
default: exit(0);
        break;
    }
  }
}
```