Lab Program - 10

Write a program
a) To construct a binary search tree
b) To traverse the tree using all the methods i.e in-order, pre order and post order
c) To display the elements in the tree

```
#include <stdio.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;
}
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);
  }
}
NODE delete(NODE root, int item)
{ NODE cur, parent, q, suc;
  if (root == NULL)
  { printf ("Tree 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;
  freenode (cur);
  return root;
}
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);
    }
  }
void main ()
{
  int item, choice;
  NODE root = NULL;
  for (;;)
{
printf ("\n1. Insert \n2. Display \n3. Pre-order \n4. Post-order
\n5. In-order \n6. Delete \n7. Exit \n");
  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: printf ("Contents of Binary Search Tree : \n");
          display (root, 0);
          break;
  case 3: printf ("Pre-order : \n");
          preorder (root);
          break;
```

```
case 4: printf("Post-order:\n");
        postorder(root);
        break;
case 5: printf("In-order:\n");
        inorder(root);
        break;
case 6: printf("Enter the item\n");
        scanf("%d", &item);
        root = delete(root, item);
        break;
case 7: exit(0);
default: printf("Invalid choice\n");
    }
  }
}
```