23/12/2020 WEEK-10-LAB-PROGRAM-10

Write a program:-

(a) To construct a binary search tree
(b) To traverse the tree using i.e, in-order, preorder and postorder.
(c) To display the elements in the tree.

CODE:

```
#include <stdio.h>
#include <conio.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);
}
}
NODE delete (NODE root, int item)
{
```

```
NODE cur, parent, q, suc;
if (root == NULL)
{
printf ("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;
```

```
freenode (ne);
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 \n4. post \n5. in \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: display(root, 0);
break;
case 3: preorder(root);
break;
case 4: postorder(root);
break;
case 5: inorder(root);
break;
case 6: printf("enter the item\n");
scanf("%d", &item);
root = delete(root, item);
break;
default: exit(0);
break;
}
}
}
```