

EXPERIMENT 7

Code:

```
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>

struct Node{
    struct Node *left;
    int data;
    struct Node *right;
}*root;

void create(){
    struct Node *nn,*temp,*par;
    nn=(struct Node*) malloc(sizeof(struct Node));
    int data;
    printf("Enter value to be added: ");
    scanf("%d",&data);
    nn->data=data;
    if(root==NULL){
        root=nn;
    }
    else{
        temp=root;
        while(temp!=NULL){
            if(data<temp->data){
                par=temp;
                temp=temp->left;
            }
            else{
                par=temp;
                temp=temp->right;
            }
        }
        if(data<par->data){
```

```

        par->left=nn;
    }
    else{
        par->right=nn;
    }
}
nn->left=NULL;
nn->right=NULL;
}

void inorderdisp(struct Node *temp){
    if(temp!=NULL){
        inorderdisp(temp->left);
        printf("%d\t",temp->data);
        inorderdisp(temp->right);
    }
}

void preorderdisp(struct Node *temp){
    if(temp!=NULL){
        printf("%d\t",temp->data);
        inorderdisp(temp->left);
        inorderdisp(temp->right);
    }
}

void postorderdisp(struct Node *temp){
    if(temp!=NULL){
        inorderdisp(temp->left);
        inorderdisp(temp->right);
        printf("%d\t",temp->data);
    }
}

void delete1(){
    struct Node *temp,*par;
    int val;
}

```

```

printf("Enter value to be deleted: ");
scanf("%d",&val);
temp=root;
while(temp!=NULL && temp->data!=val){
    if(val<temp->data){
        par=temp;
        temp=temp->left;
    }
    else{
        par=temp;
        temp=temp->right;
    }
}
if(temp==NULL)
    printf("Value not found!");
else if(temp->left==NULL&&temp->right==NULL){
    if(par->left==temp){
        par->left=NULL;
        free(temp);
    }
    else{
        par->right=NULL;
        free(temp);
    }
}
else if(temp->left!=NULL && temp->right==NULL){
    if(par->left==temp){
        par->left=temp->left;
        free(temp);
    }
    else {
        par->right=temp->left;
        free(temp);
    }
}

```

```

    }
}

else if(temp->left==NULL && temp->right!=NULL){
    if(par->left==temp){
        par->left=temp->right;
        free(temp);
    }
    else {
        par->right=temp->right;
        free(temp);
    }
}
else{
    struct Node *succ,*psucc;
    succ=temp->left;
    while(succ->right!=NULL){
        psucc=succ;
        succ=succ->left;
    }
    temp->data=succ->data;
    psucc->right=succ->left;
    free(succ);
}
}

void main(){
    int ch=0;
    while(ch!=-1){
        printf("\n1.Insert\n2.Inorder Display\n3.Preorder
Display\n4.Postorder Display\n5.Delete\n-1.Exit");
        printf("\nEnter choice: ");
        scanf("%d",&ch);
        switch(ch){
            case 1:
                create();

```

```
        break;

    case 2:
        printf("Inorder Display: ");
        inorderdisp(root);
        break;

    case 3:
        printf("Preorder Display: ");
        preorderdisp(root);
        break;

    case 4:
        printf("Postorder Display: ");
        postorderdisp(root);
        break;

    case 5:
        delete1();
        break;

    case -1:
        printf("Exiting ...");
        break;
    }
}
}
}
```

Output:

```
1.Insert
2.Inorder Display
3.Preorder Display
4.Postorder Display
5.Delete
-1.Exit
Enter choice: 1
Enter value to be added: 100
```

```
1.Insert
2.Inorder Display
3.Preorder Display
4.Postorder Display
5.Delete
-1.Exit
Enter choice: 1
Enter value to be added: 50

1.Insert
2.Inorder Display
3.Preorder Display
4.Postorder Display
5.Delete
-1.Exit
Enter choice: 1
Enter value to be added: 150

1.Insert
2.Inorder Display
3.Preorder Display
4.Postorder Display
5.Delete
-1.Exit
Enter choice: 3
Preorder Display: 100    50      150
1.Insert
2.Inorder Display
3.Preorder Display
4.Postorder Display
5.Delete
-1.Exit
Enter choice: 4
Postorder Display: 50    150      100
1.Insert
2.Inorder Display
3.Preorder Display
4.Postorder Display
5.Delete
-1.Exit
Enter choice: 1
```

```
Enter value to be added: 25
```

```
1.Insert  
2.Inorder Display  
3.Preorder Display  
4.Postorder Display  
5.Delete  
-1.Exit  
Enter choice: 5
```

```
Enter value to be deleted: 50
```

```
1.Insert  
2.Inorder Display  
3.Preorder Display  
4.Postorder Display  
5.Delete  
-1.Exit  
Enter choice: 2  
Inorder Display: 25      100      150  
1.Insert  
2.Inorder Display  
3.Preorder Display  
4.Postorder Display  
5.Delete  
-1.Exit  
Enter choice: 5  
Enter value to be deleted: 150
```

```
1.Insert  
2.Inorder Display  
3.Preorder Display  
4.Postorder Display  
5.Delete  
-1.Exit  
Enter choice: 2  
Inorder Display: 25      100  
1.Insert  
2.Inorder Display  
3.Preorder Display  
4.Postorder Display  
5.Delete  
-1.Exit  
Enter choice: -1  
Exiting ...
```