Aim:

Source Code:

BSTmain2.c

```c
#include<stdio.h>
#include<stdlib.h>
#include "BSTDeleteAndPreOrder.c"

void main() {
    int x, op;
    BSTNODE root = NULL;
    while(1)
    {
        printf("1.Insert 2.Delete 3.Preorder Traversal 4.Exit\n");
        printf("Enter your option : ");
        scanf("%d", &op);
        switch(op) {
            case 1:  printf("Enter an element to be inserted : ");
                    scanf("%d", &x);
                    root = insertNodeInBST(root,x);
                    break;
            case 2:  printf("Enter an element to be deleted : ");
                    scanf("%d", &x);
                    root = deleteNodeInBST(root,x);
                    break;
            case 3:
                    if(root == NULL) {
                        printf("Binary Search Tree is empty.\n");
                    }
                    else {
                        printf("Elements of the BST (pre-order traversal): ");
                        preorderInBST(root);
                        printf("\n");
                    }
                    break;
            case 4:  exit(0);
        }
    }
}
```

BSTDeleteAndPreOrder.c

```c
struct node {
    int data;
    struct node *left, *right;
};

typedef struct node * BSTNODE;
```

```c
    BSTNODE temp =  (BSTNODE)malloc(sizeof(struct node));
    temp->data = item;
    temp->left = temp->right = NULL;
    return temp;
}

void preorderInBST(BSTNODE root) {
    if(root!=NULL)
    {
        printf("%d ",root->data);
        preorderInBST(root->left);
        preorderInBST(root->right);
    }
}

BSTNODE insertNodeInBST(BSTNODE node, int ele) {
    if (node == NULL) {
        printf("Successfully inserted.\n");
        return newNodeInBST(ele);
    }
    if (ele < node->data)
        node->left  = insertNodeInBST(node->left,ele);
    else if (ele > node->data)
        node->right = insertNodeInBST(node->right,ele);
    else
        printf("Element already exists in BST.\n");
    return node;
}



BSTNODE minValueNode(BSTNODE node) {
    BSTNODE curr=node;
    while(curr&&curr->left!=NULL)
    {
        curr=curr->left;
    }
}

BSTNODE deleteNodeInBST(BSTNODE root, int ele){
    if(root==NULL)
    {
        printf("Cannot find %d in the binary search tree.\n");
        return root;
    }
    if(ele<root->data)
    {
        root->left=deleteNodeInBST(root->left,ele);
    }
    else if(ele>root->data)
    {
        root->right=deleteNodeInBST(root->right,ele);
    }
    else
    {
        if(root->left==NULL)
        {
```

```
        BSTNODE temp=root->right;
        printf("Deleted %d from binary search tree.\n",root->data);
        free(root);
        return temp;
    }
    else if(root->right==NULL)
    {
        BSTNODE temp=root->left;
        printf("Deleted %d from binary search tree.\n",root->data);
        free(root);
        return temp;
    }
    else
    {
        BSTNODE temp=minValueNode(root);
        root->data=temp->data;
        root->right=deleteNodeInBST(root->right,ele);
    }
  }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output
1.Insert 2.Delete 3.Preorder Traversal 4.Exit 1
Enter your option : 1
Enter an element to be inserted : 56
Successfully inserted. 2
1.Insert 2.Delete 3.Preorder Traversal 4.Exit 2
Enter your option : 2
Enter an element to be deleted : 35
Cannot find 35 in the binary search tree. 3
1.Insert 2.Delete 3.Preorder Traversal 4.Exit 3
Enter your option : 3
Elements of the BST (pre-order traversal): 56 4
1.Insert 2.Delete 3.Preorder Traversal 4.Exit 4
Enter your option : 4

Test Case - 2

User Output
1.Insert 2.Delete 3.Preorder Traversal 4.Exit 1
Enter your option : 1
Enter an element to be inserted : 25
Successfully inserted. 1
1.Insert 2.Delete 3.Preorder Traversal 4.Exit 1
Enter your option : 1
Enter an element to be inserted : 65
Successfully inserted. 2
1.Insert 2.Delete 3.Preorder Traversal 4.Exit 2
Enter your option : 2

Enter an element to be deleted : 65
Deleted 65 from binary search tree. 3
1.Insert 2.Delete 3.Preorder Traversal 4.Exit 3
Enter your option : 3
Elements of the BST (pre-order traversal): 25 4
1.Insert 2.Delete 3.Preorder Traversal 4.Exit 4
Enter your option : 4