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

typedef struct bst {
        int data;
        struct bst * left;
        struct bst * right;
}bst;

bst * root = NULL;

bst * insert() {

        int x;
        printf("Enter the element(-1 to terminate.) : ");
        scanf("%d" , &x);

        if (x == -1)
                return NULL;

        else {
                bst * newnode = (bst *)malloc(sizeof(bst));
                newnode->data = x;
                newnode->left = NULL;
                newnode->right = NULL;

                printf("Enter the data element to the left of node
%d.\n" , x);
                newnode->left = insert();

                printf("Enter the data element to the right of node
%d.\n" , x);
                newnode->right = insert();

                return newnode;
        }
}

void preorder(bst * temp) {

        if(temp != NULL) {
                printf("%d " , temp->data);
                preorder(temp->left);
                preorder(temp->right);
        }
}

void inorder(bst * temp) {

        if(temp != NULL) {
                inorder(temp->left);
                printf("%d " , temp->data);
                inorder(temp->right);
        }
}
```

```
void postorder(bst * temp) {

        if(temp != NULL) {
                postorder(temp->left);
                postorder(temp->right);
                printf("%d " , temp->data);
        }
}

void display() {
        if (root != NULL) {
                printf("Enter your operation of choice.\n");
                int ch;
                printf("1.Preorder \t 2.Inorder \t 3.Postorder  : ");
                scanf("%d" , &ch);
                switch(ch) {
                        case 1 : printf("Preorder Traversal : ");
                                        preorder(root);
                                        printf("\n");
                                        break;
                        case 2 : printf("Inorder Traversal : ");
                                        inorder(root);
                                        printf("\n");
                                        break;
                        case 3 : printf("Postorder Traversal : ");
                                        postorder(root);
                                        printf("\n");
                                        break;
                        default : printf("Invalid Input.\n");
                                         break;
                }
        }
        else
                printf("Binary Search Tree is empty.\n");
}

void main() {
        printf("Enter your operation of choice.\n");
        int ch , flag = 1;
        while(flag) {
                printf("1.Create \t 2.Display \t 3.Exit  : ");
                scanf("%d",&ch);
                switch(ch) {
                        case 1 : root = insert();
                                        break;
                        case 2 : display();
                                        break;
                        case 3 : flag = 0;
                                        break;
                        default : printf("Invalid Input.\n");
                                         break;
                }
        }
}
```