Lab Program - 10

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

struct node {
        struct node *left;
        int value;
        struct node *right,

}
typedef struct node *NODE;

NODE getNode(){
    NODE temp,
    temp = (NODE) malloc( sizeof(struct node));
    return temp,
}
~~void freeNode~~

NODE insert(NODE root){
      int value;
      NODE temp, curr, prev;
      temp = getNode();
      ~~temp~~ scanf("%d", &value);
      temp -> value = value,
      temp -> left = NULL;
      temp -> right = NULL;
      if(root == NULL){
          return temp;
      }
      prev = NULL;
      curr = root,
while (curr != NULL){
    prev = curr;
    if(value < curr -> ~~inf~~ value ){
            curr = curr -> left;

    }else{
        curr = curr -> right;
    }

}
if(value < prev -> value){
    prev -> left = temp;
}else{
    prev -> right = temp;
}
return root;
}
void preOrder(NODE root){
    if(root == NULL){
        return;
    }
    printf("%d ", root -> value);
    preOrder(root -> left);
    preOrder(root -> right);
}
void inOrder(NODE root){
    if(root == NULL){
        return;
    }
    inOrder(root -> left);
    printf(" %d ", root -> value);
    inOrder(root -> right);
}
```

```
void postOrder(NODE root){
        if(root == NULL){
                return;
        }
        postOrder(root->left);
        postOrder(root->right);
        printf("%d ", root->value);
}
void display(NODE root, int i){
      int j;
      if(root != NULL){
          display(root->right, i+1);
          for(j=0; j<i; j++){
                  printf(" ");
          printf("%d\n", root->value);
          display(root->left, i+1);
      }
}
void main(){
      int chq; NODE root = NULL;
      while(1){
          printf("Enter choice\n1- insert\n 2-display\n 3-p
                  4 - inorder\n 5- postorder\n6-exit\n");
          scanf(" %d", &chq);
          switch(chq){
              case 1 :
                  root = insert(root);
                  break;
              case 2 :
                  display(root);
                  break;
        ~~case 3~~:
            if(root == NULL){
                pf(" Tree is empty\n');
            }else{
                display(~~NODE~~root);
            }
            break;

        case 3 :
            preOrder (root);
            break;

       case 4 :
            inOrder (root);
            break;

       case 5 :
            postOrder (root);
            break;

      case 6 :
            ~~E~~, return ;

    default:
        pf("Ent~~o~~er given option \n");
        break;

rder\n  }

  }

}
```