LAB ASSIGNMENT 8
Trees: BST and Traversing algorithms

1. Write a menu program to implement Binary Search Tree (Insertion, Deletion, traversing as In-order, Pre-order and Post-order).

Solutions:

1. Write a menu program to implement Binary Search Tree (Insertion, Deletion, traversing as In-order, Pre-order and Post-order).

```
#include
<stdio.h>
struct node {
  int data;
  struct node *left, *right;
};
struct node* newNode(int item) {
  struct node* temp = (struct
  node*)malloc(sizeof(struct node)); temp->data
  = item;
  temp->left = temp->right =
  NULL; return temp;
}
struct node* insert(struct node* node,
  int key) { if (node == NULL) return
  newNode(key);
  if (key < node->data)
    node->left =
  insert(node->left, key); else if
  (key > node->data)
    node->right = insert(node->right,
  key); return node;
}
struct node* minValueNode(struct node*
  node) { struct node* current = node;
  while (current && current->left !=
```

```
    NULL) current = current->left;
  return current;
}
struct node* deleteNode(struct node* root, int key) {
```

```
  if (root == NULL) return
  root; if (key <
  root->data)
    root->left =
  deleteNode(root->left, key); else if
  (key > root->data)
    root->right =
  deleteNode(root->right, key); else {
    if (root->left == NULL) {
      struct node* temp =
      root->right; free(root);
      return temp;
    } else if (root->right ==
      NULL) { struct node*
      temp = root->left;
      free(root);
      return temp;
    }
    struct node* temp =
    minValueNode(root->right);
    root->data = temp->data;
    root->right = deleteNode(root->right, temp->data);
  }
  return root;
}
void inorder(struct node*
  root) { if (root != NULL)
  {
    inorder(root->left);
    printf("%d ",
    root->data);
    inorder(root->right)
    ;
  }
}
void preorder(struct node*
  root) { if (root != NULL)
  {
    printf("%d ",
```

```
    root->data);
    preorder(root->left)
    ;
    preorder(root->righ
    t);
  }
}
void postorder(struct node*
  root) { if (root != NULL) {
```

```
    postorder(root->left
    );
    postorder(root->rig
    ht); printf("%d ",
    root->data);
  }
}
int main() {
  struct node* root =
  NULL; int choice, val;
  while (1) {
    scanf("%d",
    &choice); if
    (choice == 1) {
      scanf("%d",
      &val); root =
      insert(root, val);
    } else if (choice == 2)
      { scanf("%d",
      &val);
      root = deleteNode(root, val);
    } else if (choice ==
      3) {
      inorder(root);
      printf("\n");
    } else if (choice == 4)
      { preorder(root);
      printf("\n");
    } else if (choice == 5)
      { postorder(root);
      printf("\n");
    } else break;
  }
  return 0;
}
```