Page No.
Date

LAB-10.

NAME: ANSHUL H. SURANA
USN: 1BM19CS020

Q-10) Write a program
a) To construct a binary search Tree
b) To traverse the tree using all the methods i.e. in order, preorder & post order.
c) To display the elements in the tree.

```
#include <stdio.h>
#include <stdlib.h>
~~struct node~~
[illegible]
    ~~struct node~~
typedef struct node
{     int data;
    struct node * left;
    struct node * right;
} Node;
void tree();
Node * Create();
Node * insert (Node *, Node*);
void traverse();
void preorder (Node*);
void inOrder (Node*);
void postOrder (Node*);
void display (Node*, int);

Node* root;
int main()
{    tree();
```

Anshul H. Surana

```
  return 0;
}
void tree()
{    int choice;
  printf(" Binary Search Tree \n  1. Insert Element
    \n   2. Traverse  All  methods \n 3. Display
      BST\n   4. Exit\n   Choice");
  scanf( " %d", & choice);
  switch (choice)
 { case 1: insert(root, create());
           break;
  case 2:    traverse();
             break;
  case 3:     if (root==NULL)
          printf(" \n  Tree  is  empty!");
           else
          display (root, 0);
           break;
 case 4:  exit(0);
           break;
 default: printf ("\n  Error Choice \n");
           tree();
  }  tree();
}
 Node *  create()
{  Node * newnode = (Node*) malloc (size of(Node));
  printf("\n  Enter  the  element:");
  scanf("%d", &newnode -> data);
  newnode -> left = NULL;
  newnode -> right = NULL;
  return newnode;
}
```

Anshul H. Surana

```
Node * insert (Node *Root, Node *newNode)
{   if (root = = NULL)
    {   root = newnode;
        printf("\n Root Node Created");
    } else
    {   if (Root → right = NULL)
        {
                Root → right = newNode;
        }
        else
        insert (Root → right, newNode);
    }
    else
    if (newNode → data < Root → data)
   {    if (Root → left = = NULL)
     {   Root → left = newNode;
     }
     else
     insert (Root → left, newNode);
    }
  }
}
void traverse()
{   if (root = = NULL)
    {   printf ("\n The tree is Empty");
        return;
    }
    printf ("\n Pre Order Traverse:");
    preOrder(root);
    printf("\n Inorder Traverse:");
    inorder (root);
```

Anshul H. Supana

NAME: ANSHUL H. SURANA
USN: 1BM19CS020

```
printf("\n Post Order Traverse:");
 PostOrder (root);
}
void PreOrder (Node * Root)
{     if (Root != NULL)
    { printf("%d", Root → data);
      preOrder (Root → left);
      preOrder ( Root → right);
    }
 }
void inorder (Node *Root)
{  if (Root != NULL)
   {   postOrder (Root → left);
       postOrder( Root → right);
       Printf(" %d ", Root → data);
       inorder (Root → right);
   }
 }
void postOrder (Node * Root)
{  if (Root != NULL)
   { PostOrder (Root → left);
     postOrder (Root → right);
     printf(" %d ", Root → data);
   }
}
void display (Node* root, int i)
{     int j;
   if (root! = NULL)
  {  display (root → right , i+1);
     for (j=0;  j< i; j++)
     Prinf(" %d \n", root → data);
     display (root → left, i+1);
  }
}
```

Anshul H. Surana