LAB 8

8) Write a program

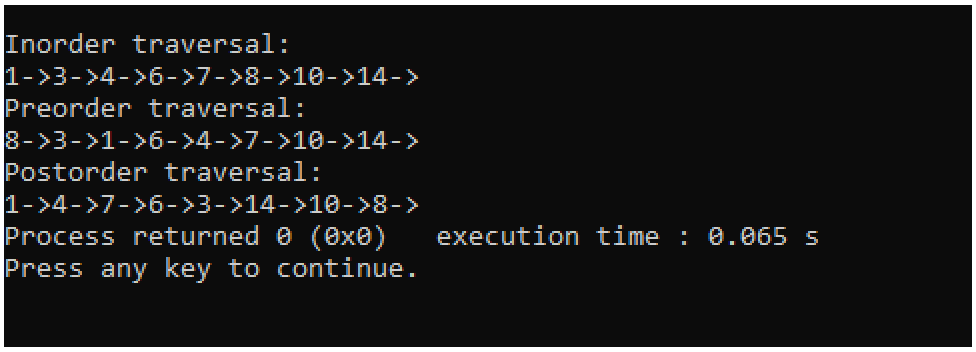
a) To construct a binary Search tree.

b) To traverse the tree using all the methods i.e., in-order, preorder and post order

c) To display the elements in the tree.

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

struct bite
{
    int data;
    struct bite* left,*right;

};

struct bite* newnode(int value)
{

    struct bite* temp= (struct bite*)malloc(sizeof(struct bite));
    temp->data=value;
    temp->left=temp->right=NULL;
    return temp;
}

struct bite* insert(struct bite* node, int value)// draw diagram
{
  if(node==NULL)//insert when the root->left or right is null(not initially)
        return newnode(value);
   if(value<node->data)
  {
      node->left=insert(node->left,value);
  }
  else if(value>node->data)
  {
      node->right=insert(node->right,value);
  }
  return node;
}

void postor(struct bite* root)
{
    if(root!=NULL)
    {
        postor(root->left);//go till the last then enter root->right which is null so prints
        postor(root->right);
        printf("%d->",root->data);
    }
}
void preor(struct bite* root)
{
    if(root!=NULL)
    {
        printf("%d->",root->data);
        preor(root->left);
        preor(root->right);
    }

}
void inor(struct bite* root)
{
    if (root != NULL)
    {
        inor(root->left);
        printf("%d->", root->data);
        inor(root->right);
    }

}
int main() {
  struct node *root = NULL;
  root = insert(root, 8);
  root = insert(root, 3);
  root = insert(root, 1);
  root = insert(root, 6);
  root = insert(root, 7);
  root = insert(root, 10);
  root = insert(root, 14);
  root = insert(root, 4);

  printf("\nInorder traversal: \n");
  inor(root);

  printf("\nPreorder traversal: \n");
  preor(root);
  printf("\nPostorder traversal: \n");
  postor(root);

}
```

OUTPUT:



```
Inorder traversal:
1->3->4->6->7->8->10->14->
Preorder traversal:
8->3->1->6->4->7->10->14->
Postorder traversal:
1->4->7->6->3->14->10->8->
Process returned 0 (0x0)   execution time : 0.065 s
Press any key to continue.
```