1. 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>
typedef struct NODE
{
   int info;
   struct NODE *lchild;
   struct NODE *rchild;
}NODE;
NODE *root=NULL;
void create();
void insert(int);
void inorder(NODE *);
void preorder(NODE *);
void postorder(NODE *);
int main()
{
   int ch,key;
   do
   {
      printf("1.create\t2.inorder\t3.preorder\t4.postorder\t5.exit\n");
      printf("Enter your choice\n");
      scanf("%d",&ch);
      switch(ch)
      {
         case 1 : create();
         break;
         case 2 : inorder(root);
         break;
         case 3 : preorder(root);
         break;
         case 4 : postorder(root);
         break;
         case 5 : exit(0);
         default : printf("Invalid choice");
      }
   }while(ch!=6);
   return 0;
}
void create()
{
   int n,i,e;
   printf("enter the number of elements\n");
   scanf("%d",&n);
   printf("enter the elements one by one\n");
   for(i=1;i<=n;i++)
   {
      scanf("%d",&e);
      insert(e);
   }
   printf("tree constructed\n");
}
```

```
void insert(int e)
{
   NODE *nn,*temp,*prev;
   nn=(NODE *)malloc(sizeof(NODE));
   nn->info=e;
   nn->lchild=NULL;
   nn->rchild=NULL;
   if(root==NULL)
   {
      root=nn;
      return;
   }
   temp=root;
   while(temp!=NULL)
   {
      prev=temp;
      if(e<temp->info)
      temp=temp->lchild;
      else if(e>temp->info)
      temp=temp->rchild;
      else
      {
         printf("its a duplicate node");
         return;
      }
   }
   if(e<prev->info)
   prev->lchild=nn;
   else
   prev->rchild=nn;
}
void inorder(NODE *tree)
{
   if(tree!=NULL)
   {
      inorder(tree->lchild);
      printf("%d\n",tree->info);
      inorder(tree->rchild);
   }
}
void preorder(NODE *tree)
{
   if(tree!=NULL){
   printf("%d\n",tree->info);
   preorder(tree->lchild);
   preorder(tree->rchild);
   }
}
void postorder(NODE *tree)
{
   if(tree!=NULL)
   {
      postorder(tree->lchild);
      postorder(tree->rchild);
      printf("%d\n",tree->info);
   }
}
```

```
1.create        2.inorder       3.preorder      4.postorder     5.exit
Enter your choice
1
enter the number of elements
5
enter the elemeents one by one
10
50
30
90
100
tree constructed
1.create        2.inorder       3.preorder      4.postorder     5.exit
Enter your choice
2
10
30
50
90
100
1.create        2.inorder       3.preorder      4.postorder     5.exit
Enter your choice
3
10
50
30
90
100
1.create        2.inorder       3.preorder      4.postorder     5.exit
Enter your choice
4
30
100
90
50
10
1.create        2.inorder       3.preorder      4.postorder     5.exit
Enter your choice
5

Process returned 0 (0x0)   execution time : 20.002 s
Press any key to continue.
```