10. Program to implement Binary tree traversal

```
#include<stdio.h>
#include<conio.h>
#include<process.h>
struct node
{
   int data;
   struct node *left;
   struct node *right;
};
struct node *createNode(int val)
{
   struct node *temp = (struct node *)malloc(sizeof(struct node));
   temp->data = val;
   temp->left = temp->right = NULL;
   return temp;
}
void inorder(struct node *root)
 {
   if (root != NULL)
   {
      inorder(root->left);
      printf("%d\t",root->data);
      inorder(root->right);
  }
}
void preorder(struct node *root)
{
  if (root != NULL)
  {
      printf("%d\t",root->data);
      preorder(root->left);
      preorder(root->right);
  }
}
void postorder(struct node *root)
 {
   if (root != NULL)
   {
      postorder(root->left);
      postorder(root->right);
      printf("%d\t",root->data);
```

```
    }
}
struct node* insertNode(struct node* node, int val)
{
    if (node == NULL) return createNode(val);
    if (val< node->data)
    node->left = insertNode(node->left, val);
    else if (val> node->data)
    node->right = insertNode(node->right, val);
    return node;
}
void main( )
{
    struct node *root = NULL;
    int ch,item,flag;
    for(;;)
    {
        printf("\n 1.INSERT\n 2.INORDER\n 3.PREORDER \n 4.POSTORDER \n 5.EXIT\n");
        printf("\n\nEnter UR choice\n");
        scanf("%d",&ch);
        switch(ch)
                {
                        case 1 :        printf("\n\nEnter the element to be inserted\n");
                                        scanf("%d",&item);
                                        root = insertNode(root,item);
                                        break;

                                case 2 :        if(root == NULL)
                                                printf("\n *****  TREE IS EMPTY ***** \n\n");
                                                 else
                                                {
                                                        printf("\n INORDER  TRAVERSAL \n\n");
                                                        inorder(root);
                                                        printf("\n");
                                                }
                                        break;
                        case 3 :        if(root == NULL)
                                        printf("\n *****  TREE IS EMPTY ***** \n\n");
                                        else
                                         {
                                            printf("\n PREORDER  TRAVERSAL \n\n");
                                            preorder(root);
                                                printf("\n");
```

```
                                }
                                break;
                        case 4 :    if(root == NULL)
                                printf("\n *****  TREE IS EMPTY ***** \n\n");
                                else
                                {
                                   printf("\n POSTORDER  TRAVERSAL \n\n");
                                        postorder(root);
                                   printf("\n");
                                }
                                break;
                        case 5 :    printf(" INVALID CHOICE\n");
                                exit(0);
                }
        }
   getch( );
}
```

11. Program to implement Binary Search

```
#include<stdio.h>
#include<conio.h>
int search( int item, int a[ ], int n)
{
        int low, high,key,mid;
        low = 0;        //Initialization
        high = n-1;     // Initialization
        key=item;
        while( low <= high )
        {
                mid = ( low + high ) / 2;               // Find the mid-point
                if ( key == a[mid] )
                {
           // If item not found, return position
                        return mid;
                }
      if( key < a[mid] )
        high = mid - 1;         // Search left side
                else
                        low = mid + 1;          // Search right side
        }
                        return -1; // Item not found
}
```

```
void main( )
{
   int i,item,a[10],n,pos;
   printf("Enter the size of an Array\n");
   scanf("%d",&n);
   printf("Enter the Array Elements\n");
   for(i=0;i<n;i++)
      scanf("%d",&a[i]);
   printf("The Array Elements are\n");
   for(i=0;i<n;i++)
      printf("%d\n",a[i]);
   printf("Enter the Element to be searched\n");
      scanf("%d",&item);
   pos=search(item,a,n);
   if(pos==-1)
      printf("Item not found\n");
   else
      printf("Item found\n");
   getch( );
}
```

12. Program to implement Selection Sort

```
#include<stdio.h>
void main( )
{
        int n,i,j,temp,a[20],pos;
        printf("Enter the number of items\n");
        scanf("%d",&n);
        printf("Enter the items to sort\n");
        for(i=0;i<n;i++)
                sacnf("%d",&a[i]);
        for(i=0;i<n-1;i++)
        {
                pos=i;
                for(j=i+1;j<n;j++)
                {
                        if(a[j]<a[pos])
                                pos=j;
                }
                temp=a[pos];
                a[pos]=a[i];
                a[i]=temp;
```

```
        }
        printf("The sorted items are\n");
        for(i=0;i<n;i++)
                printf("%d\n",a[i]);
}
```