1. Write a C program to print preorder, inorder, and postorder traversal on Binary Tree.

Code:

```
#include <stdio.h>
#include <stdlib.h>
struct node
{
  int data;
  struct node* left,* right;
};
struct node* nNode(int data)
{
  struct node* node = (struct node*)malloc(sizeof(struct node));
  node->data = data;
  node->left = NULL;
  node->right = NULL;
  return(node);
}
void printPostorder(struct node* node)
{
    if (node == NULL)
    return;
    printPostorder(node->left);
    printPostorder(node->right);
    printf("%d ", node->data);
}
void printInorder(struct node* node)
{
    if (node == NULL)
    return;
    printInorder(node->left);
    printf("%d ", node->data);
    printInorder(node->right);
}
void printPreorder(struct node* node)
{
    if (node == NULL)
    return;
    printf("%d ", node->data);
    printPreorder(node->left);
    printPreorder(node->right);
}
```

```
int main()
{
    struct node *root = nNode(9);
    root->left = nNode(8);
    root->right = nNode(7);
    root->left->left = nNode(6);
    root->left->right = nNode(1);
    printf("\nPreorder traversal of binary tree is \n");
    printPreorder(root);
    printf("\nInorder traversal of binary tree is \n");
    printInorder(root);
    printf("\nPostorder traversal of binary tree is \n");
    printPostorder(root);
    return 0;
}
```

2. Write a C program to create (or insert) and inorder traversal on Binary Search Tree.

Code:

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

typedef struct BST {
  int data;
  struct BST *lchild, *rchild;
} node;

void insert(node *, node *);
void inorder(node *);
node *search(node *, int, node **);

void main() {
  int choice;
  char ans = 'N';
  int key;
  node *new_node, *root, *tmp, *parent;
  node *get_node();
  root = NULL;

  printf("\nProgram For Binary Search Tree ");
  do {
    printf("\n1.Create");
    printf("\n2.Inorder Traversal");
    printf("\n3.Exit");
    printf("\nEnter your choice :");
    scanf("%d", &choice);

    switch (choice) {
    case 1:
      do {
        new_node = get_node();
        printf("\nEnter The Element ");
        scanf("%d", &new_node->data);

        if (root == NULL) /* Tree is not Created */
          root = new_node;
        else
          insert(root, new_node);

        printf("\nWant To enter More Elements?(y/n)");
        ans = getch();
      } while (ans == 'y');
      break;

    case 2:
     if (root == NULL)
        printf("Tree Is Not Created");
     else
     {
        printf("\nThe Inorder display : ");
        inorder(root);
     }
     break;
   }
  } while (choice != 3);
}
/*Get new Node*/
node *get_node() {
  node *temp;
  temp = (node *) malloc(sizeof(node));
  temp->lchild = NULL;
  temp->rchild = NULL;
  return temp;
}
```

```
/*This function is for creating a binary search tree*/
void insert(node *root, node *new_node) {
  if (new_node->data < root->data) {
    if (root->lchild == NULL)
      root->lchild = new_node;
    else
      insert(root->lchild, new_node);
  }

  if (new_node->data > root->data) {
    if (root->rchild == NULL)
      root->rchild = new_node;
    else
      insert(root->rchild, new_node);
  }
}
/*This function displays the tree in inorder fashion*/
void inorder(node *temp) {
  if (temp != NULL) {
    inorder(temp->lchild);
    printf("%d", temp->data);
    inorder(temp->rchild);
  }
}
```

3.Write a C program for the linear search algorithm.

Code:

```
#include <stdio.h>

void main()
{  int num;

   int i,  num_s, flag = 0;

   printf("Enter the number of elements ");
   scanf("%d", &num);
   int array[num];
   printf("Enter the elements one by one \n");
   for (i = 0; i < num; i++)
   {
       scanf("%d", &array[i]);
   }

   printf("Enter the element to be searched ");
   scanf("%d", &num_s);
   /*  Linear search begins */
   for (i = 0; i < num ; i++)
   {
       if (num_s == array[i] )
       {
           flag = 1;
           break;
       }
   }
   if (flag == 1)
       printf("Element is present in the array at position %d",i+1);
   else
       printf("Element is not present in the array\n");
}
```

4.Write a C program for binary search algorithm

Code:

```
#include <stdio.h>
int main()
{
    int i, first, last, mid, n, key, array[100];
    printf("Enter number of elements\n");
    scanf("%d",&n);
    printf("Enter %d integers\n", n);
    for(i = 0; i < n; i++)
    scanf("%d",&array[i]);
    printf("Enter value to find\n");
    scanf("%d", &key);
    first = 0;
    last = n - 1;
    mid = (first+last)/2;
    while (first <= last)
    {
        if(array[mid] <= key)
        first = mid + 1;
        else if (array[mid] == key)
        {
            printf("%d found at position %d\n", key, mid+1);
            break;

        }
        else
        last= mid - 1;
        mid = (first + last)/2;

    }
    if(first > last)
    printf("Not found! %d isn't present in the list\n", key);
    return 0;
}
```