Expt.No : 1 Date : Page No :

Expt. Name : Write a C program to print preOrder, inorder and post Order transversal on binary tree.

```
#include <stdio.h>
#include <Stdlib.h>
  Struct node
{
   int data;
   int value;
   Struct node *left;
   Struct node * right;
 }
   void inorder (Struct node * root)
  {
    if (root == =Null)
     return;
    inorder (root → left);
        printf(" %d → ", root → data);
    in order (root → right);
}
   Void pre Order (Struct node * root)
  {
    if (root = = Null)
    return;
     printf ("%d", root → data);
     pre order (root → left);
        pre Order (root → right);
  {
```

Signature.....

Expt.No : Date : Page No :

Expt. Name :

```
Void post order(Struct node *root)
{
 if(root==NULL)
  return;
  post order(root->left);
  post order(root->right);
      printf("%d->",root->data);
}
  Struct node *Create Node(value)
   Struct node *newNode = malloc(Size of(Struct node))
    new Node->data = value;
    new Node->left = NULL;
    new Node->right = NULL;

    return New Node;
  }
  Void main()
  {
    Struct node *root = create Node(1);
    root->left = Create Node(12);
    root->right = Create Node(9);
    root->left->left = Create Node(10);
    root->left->right = Create Node(15);
    root->right->left = Create Node(11);
    root->right->right = Create Node(16);
     printf("In order traversal\n");
        in order(root);
```

Signature.....

Expt.No : Date : Page No :

Expt. Name :

```
printf("\n preorder traversal \n");
    preOrder(root);

printf("\n Post order traversal \n");
    postOrder(root);
}
```

output:

Inorder traversal

10 → 12 → 15 → 1 → 11 → 9 → 16 →

pre order traversal

1 → 12 → 10 → 15 → 9 → 11 → 16 →

post order traversal

10 → 15 → 12 → 11 → 16 → 9 → 1 →

Expt.No : 2 Date : Page No :

Expt. Name : Write a C program to Create (or insert) and inorder traversal on binary search tree

```
#include<stdio.h>
#include<stdlib.h>
 Struct Node
{
 int key;
    Struct node *left, *right;
 }
  Struct node *newNode(int item)
  {
   Struct node *temp=(Struct node*)malloc(sizeof(Struct node));
   temp->key=item;
   temp->left=temp->right=NULL
   return temp;
}
  Void inorder(Struct node*root)
  {
    if(root=NULL)
    {
      inorder(root->left);
      printf("%d\n", root->key);
      inOrder(root->right);
      }
}
   Struct node.* insert(Struct node *node, int key)
{
```

Signature.....

```
if (node == NULL) return new Node(key);
if (key < node->key)
node->left = insert(node->left, key);
else if (key > node->key)
node->right = insert(node->right, key);
return node;
}
int main()
{
struct node * root = NULL
root = insert(root, 3);
insert(root, 12);
insert(root, 5);
insert(root, 43);
insert(root, 37);
insert(root, 98);
insert(root, 5);
inorder(root);
return 0;
}
```

output;

37

43

51

98

Expt.No : 3 Date : Page No :

Expt. Name : Write a C program depth first search (DFS) using a array?

```
#include <stdio.h>
int g[10][10], visited[10], n;
void DFS(int i){
int k;
printf("\n%d,i");
visited[i]=1;
for(k=0;k<n;k++){
 if(k=0;k<n;k++)
 &DFS(k);
 }
}
 Void main(){
  int i,k;
  printf("Enter number of vertices");
  Scanf("%d",&n);
  printf("\n Enter adjancency matrix of graph
  for(i=0; i<n;i++){
  for(k=0;k<n;k++){
  Scanf("%d",&g[i][k]);
  }
}
  for(i=0;i<n;i++){
  visited[i]=0;
  DFS[0);
  }
}
```

Signature.....

output:

Enter number of vertices: 6

Enter adjacency matrix of the graph : 101000

1 0 1 0 1 0
0 1 1 0 1 1
1 0 0 1 0 0
0 1 0 1 0 1
0 1 1 0 1 0
1 0 1 0 1 0

0
2
1
4
3
5

Expt.No : 4 Date : Page No :

Expt. Name : Write a C program breath first Search (BFS) using array)?

```
#include <stdio.h>
int a[20][20], q[20], visited[20], n, i, j, f=0, r=-1;
void bfs(int v){
   for (i=1; i<=n; i++){
      if (a[v][i] && !visited[i]){
         q[++r] = i;
         if(f<=r){
            visited[q[f]]=1;
            bfs (q[f++]);
         }
      }
   }
}
void main(){
   int v;
   printf("\n Enter the number of vertices");
   scanf("%d", &n);
   for(i=1; i<=n; i++){
      q[i]=0
      visited[i]=0;
}
   printf ("\n Enter graph data in matrix form:\n");
   for(i=1; i<=n; i++){
      for (j=1; j<=n; j++){
         scanf ("%d", &a[i][j]);
```

Signature.....

Expt.No : Date : Page No :

Expt. Name :

```
printf ("\n Enter the starting vertex");
Scanf ("%d", &v);
bfs(v);
printf ("\n The node which are reachable are:\n");
for(i=1;i<=n;i++){
  if (visited [i])
    printf ("%d\t", i);
  else
    printf("\n BFS is not Possible");
  }
 }
}
```

output

Enter the number of vertices : 3

Enter graph data in matrix form:

5

Enter the starting vertex: 1

BFS is not possible

Expt.No : Date : Page No :

Expt. Name :

5) Write a C program for linear search

```
#include <stdio.h>
int main()
{
    int array[100], search, c, n;
    printf("Enter number of elements in array");
    scanf("%d", &n);
    printf("Enter %d integer(s)\n", n);
    for (c=0; c<n; c++)
      scanf("%d", &array[c]);
    printf("Enter a number to search\n");
    scanf("%d", &search);
    for("%d", &search);
{
    if (array[c] == search)
      {
        printf("%d isn't present in array");
        return 0;
      }
```

Output

```
Enter number of Elements in array.
5
Enter 5 Integers
25
14
36
95
38
Enter a number to Search
95
95 present at location 4
```

Write a C program for binary search algorithm

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

     break;
}
```

```
    else
        last = middle - 1;
        middle = (first + last)/2;
    }
    if (first > last)
        printf("not found ! %d isn't present
                                the list \n", search);

    return 0;

}
```

end

output:

```
Enter number of Elements
5

Enter 5 integers
25
36
41
51
95

Enter Value to find in 5,
51 found at location 4
```