LAB PROGRAMS-2

M. Nitheesh
AP19110010326
CSE-F

1 //c program for different tree traversals.

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

{
int data;
struct node* left;
struct node* right;
};
{
struct node* node = (struct node*)

    node->data = data;
    node->left = NULL;
    node->right = NULL;

    return (node);
}
{
If (node == NULL)
  return;
}
void PrintInorder (struct node* node)
{
if (node == NULL)
   return;
Printf ("%d", node->data);
Print preorder (node->left);
Print preorder (node->right);
}
int main();
{
Printf ("preorder transuversal");
Print Preorder (root);
```

```
printf("\nInorder traversal\n");
printinorder(root);
printf("\n Postorder traversal\n");
printpostorder(root);

getchar();
return 0;
}
```

2) C Program to construct a Binary search tree a
Perform deletion and inorder transversal.

```
#include <stdio.h>
#include <stdlib.h>
struct btnode
{
int value;
struct btnode *l;
struct btnode *r;
} *root=NULL, *temp=NULL, *t2, *t1;

void delete1();
void insert();
void delete();
void inorder(struct btnode *t);
void create();
void search(struct btnode *t);
~~void~~ p
int flag=1;
void main()
{
int ch;
printf("\n OPERATIONS");
```

```
printf ("\n 1. Insert an element into tree");
printf ("\n 2. Delete an element from tree");
Printf ("3. Inorder Transversal");
Printf ("4. Exit");
while(1)
{
Printf ("Enter your choice: ");
scanf ("%d", &ch);
switch (ch)
{
case 1:
  insert();
  break;
case 2:
  delete();
  break;
case 3:
  inorder(root);
  break;
case 4:
  exit(0);
default:
  Printf ("wrong choice")
  break;
}
}
/* To insert a node in tree */
void insert
```

3) DFS C Program using array.

```
#include <stdio.h>
#include <conio.h>
int a[20][20], reach[20], n;
void dfs(int v){
    int i;
    reach[v]=1;
    for(i=1; i<=n; i++)
      if(a[v][i] && !reach[i]){
        printf("\n %d -> %d", v, i);
        dfs(i);
      }

void main(){
    int i, j, count = 0;
    clrscr();
    printf("Enter number");
    scanf("%d", &n);
    for(i=1; i<=n; i++){
      reach[i] = 0;
      for(j=1; j<=n; i++)
        a[i][j]=0;
    }

for(i=1; i<=n; i++)
for(j=1; j<=n; j++)
  scanf("%d", &a[i][j]);
dfs(1);
printf("\n");
for(i=1; i<=n; i++){
    if(reach[i])
      count++
}
```

4) BFS C program 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("Enter the number of vertices");
    scanf("%d", &n);
    for(i=1; i<=n; i++)
    {
        q[i]=0;
        visited[i]=0;
    }
    printf("Enter the starting vertex");
    scanf("%d", &v);
    bfs(v);
    for(i=1; i<=n; i++)
    {
        if(visited[i])
            printf("%d\t", i);
        else
        {
            printf("Bfs is not");
            break;
        }
    }
```

5. Linear search program in C

```
#include <stdio.h>
int main()
{
int array[100], search, c, n;
Printf("Enter number of elements in array\n");
scanf("%d", &n);
printf("Enter %d integer\n", n);
for (c=0; c<n; c++)
  scanf("%d", &array[c]);
Print f("Enter a no. of to search");
Scan f("%d", &search);
for (c=0; c<n; c++)
{
if (array[c] == search)
{
Print f("%d, seach, c+1);
break;
}
}
if (c == n)
printf("%d", search);
return 0;
}
```

6. Binary search in C.

```
#include <stdio.h>
int main()
{
int c, first, last, middle, n, search, array[100];
Printf("Enter no. of elements");
Scanf("%d", &n);
Printf("Enter %d integers \n", n);
for (c=0; c< n; c++)
Scanf("%d", &array[c]);
Printf("Enter Value to find");
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",
search, middle+1);
break;
}
else
last = middle-1
middle = (first+last)/2;
if (first > last)
Printf("%d isn't present in the list.\n", search);
return 0;
}
```