8. WAP for Breadth First Search :

```
#include <stdio.h>
void bfs(int a[10][10], int n, int u)
  {
    int f, r, q[10], v;
    int s[10] = {0};
    printf("the nodes visited from %d:", u);
    f=0;
    r=-1;
    q[++r] = u;
    s[u] = 1;
    printf("%d", u);
    while(f<=r)
      {
        u = q[f++];
        for( v=0; v<n; v++)
        {
        if (a[u][v])
          {
          if (s[v] == 0)
            {
            printf("%d", v);
            s[v] = 1
            q[++r] = v;
          }
          }
        }
      }
    printf("\n");
  }
int main ()
```

```
int n, a[10][10], source, i, j;
printf("\nenter no. of nodes ");
scanf("%d", &n);
printf("\nEnter the adjacency matrix ");
   for (i=0; i<n; i++)
    {
     for (j=0; j<n; j++)
      {
       scanf("%d", &a[i][j]);
      }
    }
for (source = 0; source <n; source ++)
   bfs( a, n, source );
  return 0;
}
```

op:

```
Enter no. of nodes : 7
Enter adjacency matrix : 0 1 1 1 1 0 0
1 0 0 1 0 1 0
1 0 0 0 0 0 1
1 1 0 0 0 1 0
1 0 0 0 0 0 1
0 1 0 1 0 0 0
0 0 1 0 1 0 0
The nodes visited from 0: 0 1 2 3 4 5 6
                       1: 1 0 3 5 2 4 6
                       2: 2 0 6 1 3 4 5
                       3: 3 0 1 5 2 4 6
                       4: 4 0 6 1 2 3 5
                       5: 5 1 3 0 2 4 6
                       6: 6 2 4 0 1 3 5
```

8. WAP for Depth First Search

```
#include <stdio.h>
void bfs (int a[10][10], int n, int u)
{
int f, r, q[10], v;
int s[10] = {0};
printf (" the nodes visited
```

WAP for Depth First Search.

```
#include <stdio.h>
#include <conio.h>
int a[1][10];
void dfs (int n, int cost[10][10], int u, int s[])
{
int v;
s[u] = 1;
for(v=0; v<n; v++)
{
if ((cost[u][v] == 1) && (s[v] == 0))
dfs(n, cost, v, s);
}
}
void main ()
{
int n, i, j, cost[10][10], s[10], con, flag;
printf ("Enter the no. of nodes \n");
scanf ("%d", &n);
printf ("Enter the adjacency matrix");
for (i=0; i<n; i++)
{
for (j=0; j<n; j++)
scanf ("%d", &cost[i][j]);
}
```

```
con=0;
for(j=0;j<n;j++)
{
for(i=0;i<n;i++)
s[i]=0;
dfs(n,cost,j,s);
flag=0;
for(i=0;i<n;i++)
{
if(s[i]==0)
flag=1;
}
if(flag==0)
con=-1;
}
if(con==1)
printf("Graph is connected\n");
else
printf("Graph is not connected\n");
getch();
}
```

O/P: Enter the number of nodes : 4
Enter the adjacency matrix
0 1 0 0
0 0 1 0
0 0 0 1
1 0 0 0
Graph is connected.

26.02.24