Design, Develop and Implement a Program in C for the following operations on Graph(G) of Cities
a. Create a Graph of N cities using Adjacency Matrix.
b. Print all the nodes reachable from a given starting node in a digraph using BFS method
c. Check whether a given graph is connected or not using DFS method.

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#define MAX 20

int G[MAX][MAX], n,visited1[10],visited[10];

/********Graph creation********/
void Creation_Graph()

{
 printf("Enter the Number of Cities");
scanf("%d", &n);
printf("Enter the edges for the  Cities");
        for(int i=0;i<n;i++)
        {
        for(int j=0;j<n;j++)
                {
                scanf("%d",&G[i][j]);
                }
        }
printf("Adjacency matrix for the Cities is created succesfully\n");

}

/*********Reachability**********/

void Reachability(int source)
{
        int f,r,q[10];    // for queue operations
        int u,v;          // represent two vertices
        int i;            // index variable
        f=0;
        r=-1;


        q[++r]=source;          // insert vertex into queue

        visited1[source]=1;     // add source to v (indicating source is visited)

        while(f<=r)     // as long as queue is not empty
        {
                u=q[f++];  // delete the next vertex to be explored from q
```

```
                for(v=1;v<=n;v++)
                {
// find the nodes v which are adjecent to u
                        if(visited1[v]==0 && G[u][v]==1)
                        {
                                visited1[v]=1; // add v to s indicates that v is visited now
                                r++;
                                q[r]=v; // insert new vertex into q for exploration
                        }
                }
        }

for(i=1;i<=n;i++)
        {
                if(visited1[i]==0)
                {
                        printf("vertex %d is not reachable\n",i);
                }
                else
                        printf("vertex %d is reachable\n",i);
        }


        }

 /********Connectivity***********/

                // to insert the verices which are visited

void  Connectivity(int source)
{  int visited[10];
        int v,flag=1;
        visited[source]=1;
        for(v=1;v<=n;v++)
        {
                if(visited[v]==0 && G[source][v]==1)
        Connectivity(v);
        }
for(int i=1;i<=n;i++)
        {
          if(visited[i]==0)
                flag = 0;
        }

        if(flag)
                printf("\n Graph is connected.....\n");
        else
                printf("\n Graph is not connected.....\n");
}
```

```
void main()
{
  int choice,source;
        clrscr();
        while(1)
        {

                printf("\n\n\n\t1.create a graph .\t2.Test for reachability..\t3.Test for
connectivity.....\t4.Exit...");
                printf("\n\n\n\tEnter Your Choice: ");
                scanf("%d",&choice);
                switch(choice)
                {
                        case 1:Creation_Graph();break;
                        case 2:printf(" enter the source between 0 to %d\n",n-1 );
                                scanf("%d",&source);
                                Reachability(source);break;

                        case 3: printf(" enter the source between 0 to %d\n",n-1 );
                                                scanf("%d",&source);
                                                Connectivity(source);break;
                        case 4:exit(0);
                        default: printf("\n\n\n\tEnter proper Choice....");
                }
        }

}
```