**Slip 7**

Q-1) Write a program for the Implementation of Dijkstra’s algorithm to find shortest path to other vertices

#include<iostream>

using namespace std;

void dij(int, int [20][20], int [20],int [20],int);

int main()

{

    int i,j,n, visited[20],source,cost[20][20],d[20];

    cout<<"Enter no. of vertices:";

    cin>>n;

    cout<<"Enter the cost adjacency matrix\n";

    for(i=1;i<=n;i++)

    {

        for(j=1;j<=n;j++)

        {

            cin>>cost[i][j];

        }

    }

    cout<<"\nEnter the source node:";

    cin>>source;

    dij(source, cost, visited,d,n);

    for(i=1;i<=n;i++)

        if(i!=source)

    cout<<"\nShortest path from"<<source<<"to"<<i<<"is"<<d[i];

}

void dij(int source, int cost[20][20],int visited[20],int d[20],int n)

{

    int i,j,min,u,w;

    for(i=1;i<=n;i++)

    {

        visited[i]=0;

        d[i]=cost[source][i];

    }

    visited[source]=1;

    d[source]=0;

    for(j=2;j<=n;j++)

    {

        min=999;

        for(i=1;i<=n;i++)

        {

            if(!visited[i])

            {

                if(d[i]<min)

                {

                    min=d[i];

                    u=i;

                }

            }

        }

        visited[u]=1;

        for(w=1;w<=n; w++)

        {

            if(cost[u][w]!=999 && visited[w]==0)

            {

                if(d[w]>cost[u][w]+d[u])

                d[w]=cost[u][w]+d[u];

            }

        }

    }

}

Q.2) Write a program for finding Topological sorting for Directed Acyclic Graph (DAG)

#include<iostream>

using namespace std;

void findindegree(int [10][10],int[10],int); void topological(int,int [10][10]);

int main()

{

    int a[10][10],i,j,n;

    cout<<"Enter the number of nodes:";

    cin>>n;

    cout<<"\nEnter the adjacency matrix\n";

    for(i=1;i<=n;i++)

        for(j=1;j<=n;j++)

            cin>>a[i][j];

    cout<<"\nThe adjacency matirx is:\n";

    for(i=1;i<=n;i++)

    {

        for(j=1;j<=n;j++)

        {

            cout<<a[i][j]<<"\t";

        }

        cout<<"\n";

    }

    topological(n,a);

}

void findindegree(int a[10][10], int indegree[10],int n)

{

    int i,j,sum;

    for(j=1;j<=n;j++)

    {

        sum=0;

        for(i=1;i<=n;i++)

        {

            sum = sum + a[i][j];

        }

        indegree[j] = sum;

    }

}

void topological(int n,int a[10][10])

{

    int k,top,t[100],i,stack[20],u,v,indegree[20];

    k=1;

    top=-1;

    findindegree(a,indegree,n);

    for(i=1;i<=n;i++)

    {

        if(indegree[i]=0)

        {

            stack[++top]=i;

        }

    }

    while(top!=-1)

    {

        u=stack[top--];

        t[k++]=u;

        for(v=1;v<=n;v++)

        if(a[u][v]==1)

        {

            indegree[v]--;

            if (indegree[v]=0)

            {

                stack[++top]=v;

            }

        }

    }

    cout<<"\nTopological sequence is\n";

    for(i=1;i<=n;i++)

    cout<<t[i]<<"\t";

}