Name – Saikat Sheet

University Roll No – 18700120024

Department – CSE(A)

Assignment-9

Code:

#include<stdio.h>

#include<stdlib.h>

int d[10];

int p[10];

int n,s,i,j,u,v;

int count;

int a[10][10];

void relax(int u, int v)

{

  if(d[v] > d[u] +a[u][v])

  {

    d[v] = d[u] +a[u][v];

    p[v] = u;

   }

}

void initialize\_Single\_Source()

{

   count = n;

   for(i = 0; i < n; i++ )

   {

     d[i] = 999;

     p[i] = -1;

   }

   d[s] = 0;

}

int bellmanford()

{

   initialize\_Single\_Source();

   for(i = 0; i < n; i++ )

   {

     for(u = 0; u < n; u++)

     {

       for(v = 0; v < n; v++)

       {

         if(a[u][v] != 999)

         {

           relax(u, v);

         }

       }

      }

 }

 for(u = 0; u < n; u++)

 {

   for(v = 0; v < n; v++)

   {

      if(a[u][v] != 999)

      {

        if(d[v] > d[u]+a[u][v])

         return 0;

      }

   }

 }

 return 1;

}

int main()

{

  printf("Enter the number of nodes: ");

  scanf("%d", &n);

  printf("Enter the matrix\n");

  for(i = 0; i < n; i++)

  {

    for(j = 0; j < n; j++)

    {

      scanf("%d", &a[i][j]);

    }

  }

  printf("Enter start node: ");

  scanf("%d", &s);

  int true = bellmanford();

  if(true)

   {

       printf("Shortest path is possible\nThe shortest path for each nodes are: \n");

       for(i = 0; i < n; i++)

       {

         printf("d[%d] = %d, p[%d] = %d.\n", i, d[i], i, p[i]);

       }

   }

   else

   printf("Shortest path is not possible\n");

   return 0;

}

