***Experiment***

# Aim: Implement the Bellman Ford Algorithm used for designing distance vector routing protocol.

# Theory: Routing algorithm is a part of network layer software which is responsible for deciding which output line an incoming packet should be transmitted on. If the subnet uses datagram internally, this decision must be made anew for every arriving data packet since the best route may have changed since last time. If the subnet uses virtual circuits internally, routing decisions are made only when a new established route is being set up. The latter case is sometimes called session routing, because a rout remains in force for an entire user session (e.g., login session at a terminal or a file).

# Code:

# #include<bits/stdc++.h>

# using namespace std;

# struct node {

# int ds[20];

# int reach[20];

# } re[10];

# int main()

# {

# int dis[20][20], n;

# cout << "Number of nodes : " << endl;

# cin >> n;

# cout << "Distance matrix : " << endl;

# for (int i = 0; i < n; i++) {

# for (int j = 0; j < n; j++) {

# cin >> dis[i][j];

# dis[i][i] = 0;

# re[i].ds[j] = dis[i][j];

# re[i].reach[j] = j;

# }

# }

# int flag;

# do {

# flag = 0;

# for (int i = 0; i < n; i++) {

# for (int j = 0; j < n; j++) {

# for (int k = 0; k < n; k++) {

# if ((re[i].ds[j]) > (re[i].ds[k] + re[k].ds[j])) {

# re[i].ds[j] = re[i].ds[k] + re[k].ds[j];

# re[i].reach[j] = k;

# flag = 1;

# }

# }

# }

# }

# } while (flag);

# for (int i = 0; i < n; i++) {

# cout << "Router info : " << i + 1 << endl;

# cout << "Dest\t Next Hop\t Distance" << endl;

# for (int j = 0; j < n; j++)

# printf("%d\t\t%d\t\t%d\n", j+1, re[i].reach[j]+1, re[i].ds[j]);

# }

# return 0;

# }

# Result:

# 