Experiment – 6

AIM- To implement distance vector routing algorithm.

Description - A distance-vector routing (DVR) protocol requires that a router inform its neighbors of topology changes periodically. Historically known as the old ARPANET routing algorithm (or known as Bellman-Ford algorithm).

Algorithms:

1. A router transmits its distance vector to each of its neighbors in a routing packet.
2. Each router receives and saves the most recently received distance vector from each of its neighbors.
3. A router recalculates its distance vector when:
   a) It receives a distance vector from a neighbor containing different information that before.
   b) It discovers that a link to a neighbor has gone down.

The DV calculation is based on minimizing the cost to each destination

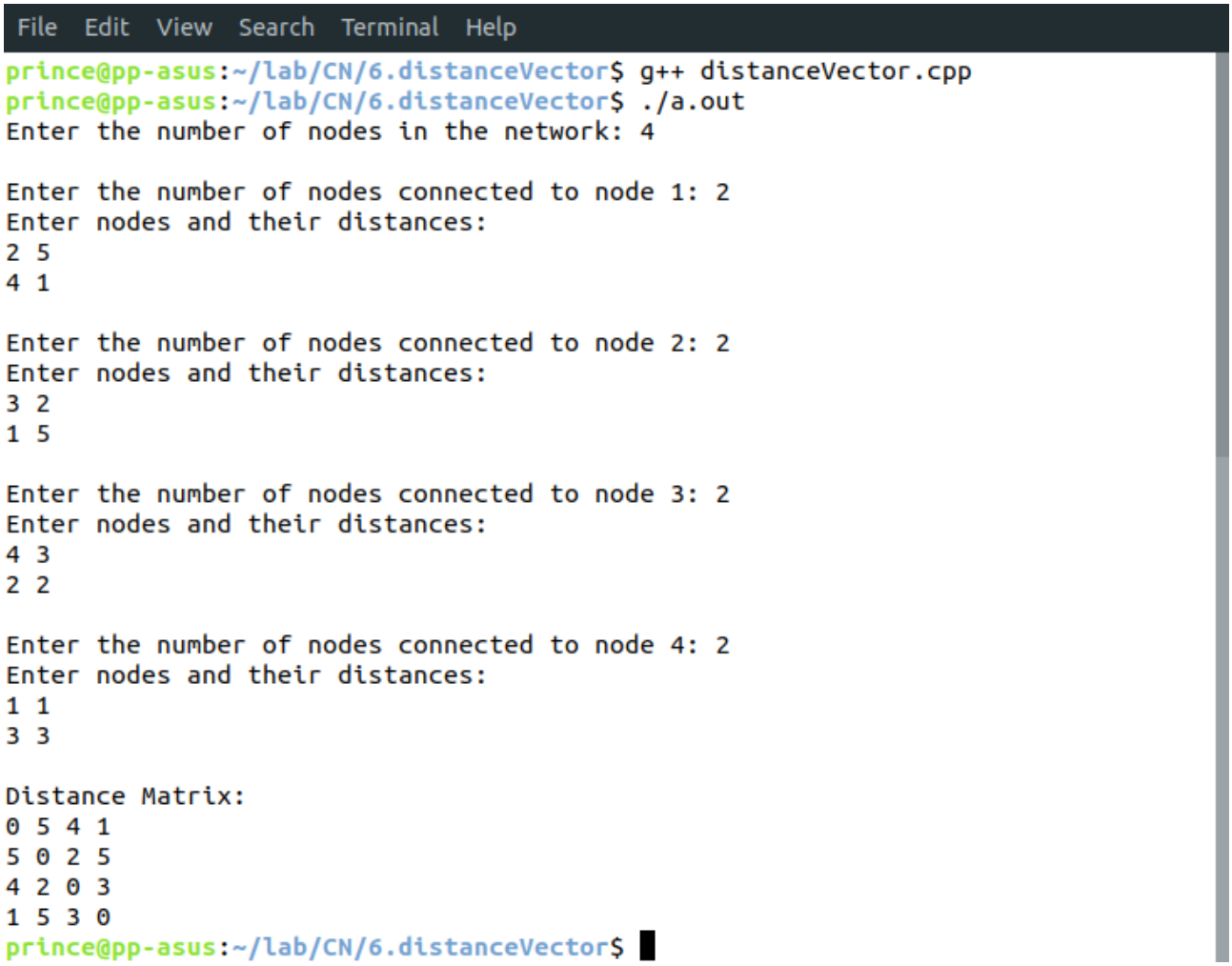
Dx(y) = Estimate of least cost from x to y
C(x,v) = Node x knows cost to each neighbor v
Dx = [Dx(y): y ∈ N] = Node x maintains distance vector
Node x also maintains its neighbors' distance vectors
– For each neighbor v, x maintains Dv = [Dv(y): y ∈ N]

Code:

```
#include<bits/stdc++.h>
using namespace std;

int main(){
    int n;
    cout<<"Enter the number of nodes in the network: ";
    cin>>n;

    vector<vector<int>> distTable(n+1, vector<int> (n+1, 10000));
    for(int i=1; i<=n; i++){
        int childs;
        cout<<endl;
```

```
        cout<<"Enter the number of nodes connected to node "<<i<<": ";
        cin >> childs;
        distTable[i][i] = 0;
        if(childs>0){
            cout<<"Enter nodes and their distances: "<<endl;
        }
        for(int j = 0; j < childs; ++j){
            int sibling;
            int distance;
            cin >> sibling >> distance;
            distTable[i][sibling] = distance;
            distTable[sibling][i] = distance;
        }
    }

    bool done = false;
    while(!done){
        done = true;
        for(int i=1; i<=n; i++) {
            for(int j=1; j<=n ; j++){
                for(int k=1; k<=n; k++){
                    if(distTable[i][j] > (distTable[i][k]+distTable[k][j])){
                        distTable[i][j] = distTable[i][k] + distTable[k][j];
                        done = false;
                    }
                }
            }
        }
    }
    cout<<endl;
    cout<<"Distance Matrix: "<<endl;
    for(int i=1; i<=n ; i++){
        for (int j=1; j<=n ; j++){
            if(distTable[i][j] == 10000)
                distTable[i][j] = -1;
            cout<<distTable[i][j]<<" ";
        }
        cout<<endl;
    }

    return 0;
}
```

Output:



Learnings –

1. **Advantages of Distance Vector routing** –
   - It is simpler to configure and maintain than link state routing.
2. **Disadvantages of Distance Vector routing** –
   - It is slower to converge than link state.
   - It is at risk from the count-to-infinity problem.