// CPP program to count number of shortest

// paths from a given source to every other

// vertex using BFS.

void BFS(vector<int> adj[], int src, int dist[],

                           int paths[], int n)

{

    bool visited[n];

    for (int i = 0; i < n; i++)

        visited[i] = false;

    dist[src] = 0;

    paths[src] = 1;

    queue <int> q;

    q.push(src);

    visited[src] = true;

    while (!q.empty())

    {

        int curr = q.front();

        q.pop();

        for (auto x : adj[curr])

        {

            if (visited[x] == false)

            {

                q.push(x);

                visited[x] = true;

            }

            if (dist[x] > dist[curr] + 1)

            {

                dist[x] = dist[curr] + 1;

                paths[x] = paths[curr];

            }

            else if (dist[x] == dist[curr] + 1)

                paths[x] += paths[curr];

        }

    }

}

void findShortestPaths(vector<int> adj[],

                       int s, int n)

{

    int dist[n], paths[n];

    for (int i = 0; i < n; i++)

        dist[i] = INT\_MAX;

    for (int i = 0; i < n; i++)

        paths[i] = 0;

    BFS(adj, s, dist, paths, n);

    cout << "Numbers of shortest Paths are: ";

    for (int i = 0; i < n; i++)

        cout << paths[i] << " ";

}

 void addEdge(vector<int> adj[], int u, int v)

{

    adj[u].push\_back(v);

}