#include <iostream> #include <vector> #include <queue> #include <climits>

using namespace std;

int networkDelayTime(vector<vector<int>>& times, int n, int k) {

vector<int> dist(n+1, INT\_MAX); // initialize all distances to infinity dist[k] = 0; // distance to starting node is 0

priority\_queue<pair<int,int>, vector<pair<int,int>>, greater<pair<int,int>>> pq; pq.push(make\_pair(0, k)); // push starting node to priority queue

while(!pq.empty()) {

int u = pq.top().second; pq.pop(); for(auto edge : times) {

int v = edge[1], w = edge[2];

if(edge[0] == u && dist[u] + w < dist[v]) { dist[v] = dist[u] + w; pq.push(make\_pair(dist[v], v));

}

}

}

int maxDist = 0;

for(int i=1; i<=n; i++) {

if(dist[i] == INT\_MAX) return -1; // some nodes are unreachable maxDist = max(maxDist, dist[i]);

}

return maxDist;

}

int main() {

int n = 4, k = 2;

vector<vector<int>> times {{1,2,1},{2,3,2},{1,3,4},{2,4,3},{3,4,5}}; int minTime = networkDelayTime(times, n, k);

cout << "Minimum time it takes for all nodes to receive the signal: " << minTime << endl; return 0;

}