AHMED SWIDAN – 5588
MOSTAFA TALAAT - 5401

Problem 1

DESCRIPTION

1. Overview

LinkedIn is a professional networking site, people are connected with other people. The whole system appears as a giant connected graph, and when you open any user profile you can see if this user is a connection from 1st, 2nd, or 3rd connection. 1st means I'm directly connected with this user, 2nd means I'm not directly connected with this user, but I'm connected with someone who is connected with this user. 3rd means I'm not directly connected with this user, but I'm connected with someone who is connected with someone that is connected with this user. And so on.

2. Required

It's required to get the total number of people connected at kth vertices away from each other.

3. Input

Input will be number of people which will be represented using vertices and connections between them which will be represented as edges of the graph, then the starting vertex and value k.

4. Output

Number of people that are k vertices away from starting vertex.

SOLUTION

We have people and connections between them. These connections will be represented as a graph, where every human is a vertex and every connection is an edge.

Breadth first traversal will be applied where a tree will be created and every level of this tree is the level of connection (k) with respect to the root node. A flag will be initiated declaring the start of a new level and when the flag is popped out of the BFS queue then that denotes that the level has ended and a new level will begin.

DATA STRUCTURES AND RUNNING TIME

- Graph with linked list implementation for the people and the connections
- Queue for BFS

Since the implementation is of a typical graph the space and time complexity will be the same

Regarding space complexity, the complexity is O (V+E) where the worst case is O (V^2)

Regarding time complexity, Breadth-first search has a running time of O (V + E) since every vertex and every edge will be checked once. Depending on the input to the graph, O (E) could be between O (1) and O (V^2)

SAMPLE RUNS

```
please enter number of vertices:
9
please enter number of edges
10
please enter the edges in the form (s,e)
1 2
2 3
1 7
2 4
4 7
7 8
3 4
7 6
5 6
9 7
please enter the starting vertix:
4
please enter k:
2
number of elements in level 2 are 4
```

```
please enter number of vertices:
6
please enter number of edges
5
please enter the edges in the form (s,e)
1 2
2 3
3 4
4 5
5 6
please enter the starting vertix:
1
please enter k:
4
number of elements in level 4 are 1
```

```
please enter number of vertices:
6
please enter number of edges
5
please enter the edges in the form (s,e)
1 2
1 3
1 4
1 5
1 6
please enter the starting vertix:
1
please enter k:
1
number of elements in level 1 are 5
```

Problem 2

DESCRIPTION

1. Overview

A runner wants to escape out of a maze. The maze consists of (N×N) cells. Some of cells contained blocks, so, runner will avoid those cells to escape. Runner needs to know the path to follow which he can take to escape the maze.

2. Required

It's required to get the path that runner can follow to escape out of maze.

3. Input

The first line of input contains an integer "N", (i.e. the size of the (N×N) matrix). The next n lines each line contains N space-separated values either 0 or 1.

4. Output

Print the path that runner can follow to escape out of maze. In case no possible path found print "no path found".

SOLUTION

The maze is a 2D array but it's worked on as a graph where if the neighboring point equals zero then it's as if there is an edge between them, and if it's one then there is no edge.

We will apply Dijkstra's Algorithm in our program to determine the shortest paths from (0, 0) to (n-1, n-1) storing the pervious node and the cost of the trip starting from (0, 0) in each node. Then using a stack we can get the actual path.

DATA STRUCTURES AND RUNNING TIME

- A 2d array for the maze
- A 2d array for storing all the info (weight and pervious node)
- Stack for getting the actual path

Space complexity is concerned with the 2 arrays where the first one is N^2 and the second is 2N^2. So, space complexity is O(N^2)

There are 2 loops, the first one is for passing on all the nodes so it's complexity is O (N^2). The second one is to get the next lowest weighted node which runs N^2 times. That makes the time complexity O (N^4)

Note that: N is the input integer, not the number of elements. If the complexity was calculated by the number of elements then:

Time complexity would be → O (n^2)

Space complexity would be → O (n)

SAMPLE RUNS

```
please enter N:
4
please enter values for maze:
0 1 1 0
0 0 1 0
0 0 0 0
0 1 1 0
(0,0) -> (1,0) -> (1,1) -> (2,1) -> (2,2) -> (2,3) -> (3,3)
```

```
please enter N:
3
please enter values for maze:
0 1 0
0 1 0
0 1 0
no path found
```

```
please enter N:
10
please enter values for maze:
0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 0
0 0 0 0 0 0 0 0 0 0
0 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 0
0 1 0 0 0 0 0 0 0 0
1 1 0 1 1 1 1 1 1 1
1 1 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 0
(0,0) -> (0,1) -> (0,2) -> (0,3) -> (0,4) -> (0,5) -> (0,6) -> (0,7) -> (0,8) -> (0,9) -> (1,9) -> (2,9)
 -> (2,8) -> (2,7) -> (2,6) -> (2,5) -> (2,4) -> (2,3) -> (2,2) -> (2,1) -> (2,0) -> (3,0) -> (4,0) -> (
4,1) -> (4,2) -> (4,3) -> (4,4) -> (4,5) -> (4,6) -> (4,7) -> (4,8) -> (4,9) -> (5,9) -> (6,9) -> (6,8)
-> (6,7) -> (6,6) -> (6,5) -> (6,4) -> (6,3) -> (6,2) -> (7,2) -> (8,2) -> (8,3) -> (8,4) -> (8,5) -> (8
,6) -> (8,7) -> (8,8) -> (8,9) -> (9,9) ->
```

Problem 3

DESCRIPTION

1. Overview

There are some cities and some routes connecting specific cities (not all cities are connected). For each route between two cities there is a flight with specific time and cost (same for any direction). An Employee wants to travel from city X to city Y and he needs to minimize the cost that will be paid. Every hour that the employee spends during traveling or waiting in the airport for another flight connection, he has to pay M Dollars Assume that layover time between connecting flights is always one hour.

2. Required

It's required to get the path with the minimum cost for the employee during his journey.

3. Input

- Amount M that he will lose per hour.
- Number of cities.
- Number of existing routes.
- Cost and time for each flight between two cities.
- Source and destination Cities.

4. Output

The route with the minimum cost with total time and cost.

SOLUTION

We have numberOfCities which defines our 2D vector representation size, we have routes each route have info(source, destination, hours and cost) which we can store the needed values(hours and cost) into a pair, then store them in our 2D vector in their correct locations, since our graph is undirected, we will store them in both locations [source][destination] and [destination][source].

We will apply Dijkstra's Algorithm in our program to determine the shortest paths of our graph from the give source while storing the routes of our desired path from source to destination, and calculate the number of hours the whole trip took and add the given M amount for each hour to our initial cost of each route taken.

DATA STRUCTURES AND RUNNING TIME

- Vectors for the graph representation
- Pairs for storing routes
- Arrays

The program has n cities and we are using 2D vector representation which takes O(n^2) space and use 2 iterative nested loops inside eachother, each consumes O(n) time,

The outer used for getting shortest path for all given vertices, while the other used for relaxation process which re-calculate the cost for each path. This sums up to time of O(n^2).

Time complexity would be → O (n^2)

Space complexity would be → O (n^2)

SAMPLE RUNS

```
please enter amount M: 200
please enter number of cities: 2
please enter number of routes: 1
please enter source, destination, time and cost for each route:
1 2 3 500
please enter source city: 1
please enter destination city: 2
Route with minimum cost is 1 -> 2
Total time is 3
Total cost is $2100
```

```
please enter amount M: 100
please enter number of cities: 4
please enter number of routes: 5
please enter source, destination, time and cost for each route:
1 2 1 250
1 3 1 300
1 4 2 700
2 4 1 300
3 4 1 200
please enter source city: 2
please enter destination city: 3
Route with minimum cost is 2 -> 4 -> 3
Total time is 3
Total cost is $800
```

```
please enter amount M: 200
please enter number of cities: 3
please enter number of routes: 3
please enter source, destination, time and cost for each route:
1 2 2 100
1 3 3 50
2 3 1 25
please enter source city: 1
please enter destination city: 3
Route with minimum cost is 1 -> 3
Total time is 3
Total cost is $750
```

Division of labor

We both worked together in all problems.