Experiment 5

Graph

Student Name: Bhanu Pundir
UID: 20BCS1439
Branch: BE CSE
Section/Group: 620-B
Semester: 5th
Date of Performance: 07 Oct 2022
Subject Name: CC Lab
Subject Code: 20CSP-314

1. Aim/Overview of the practical:

To implement the concept of Graphs.

Consider an undirected graph where each edge weighs 6 units. Each of the nodes is labeled consecutively from 1 to n.
You will be given a number of queries. For each query, you will be given a list of edges describing an undirected graph. After you create a representation of the graph, you must determine and report the shortest distance to each of the other nodes from a given starting position using the *breadth-first search* algorithm (BFS). Return an array of distances from the start node in node number order. If a node is unreachable, return for that node.
https://www.hackerrank.com/challenges/bfsshortreach/problem?isFullScreen=true

2. Apparatus / Simulator Used:

- Windows 7 or above
- Google Chrome

3. Objective:

- To understand the concept of graphs.
- To implement the concept of Graphs.

4. Code:

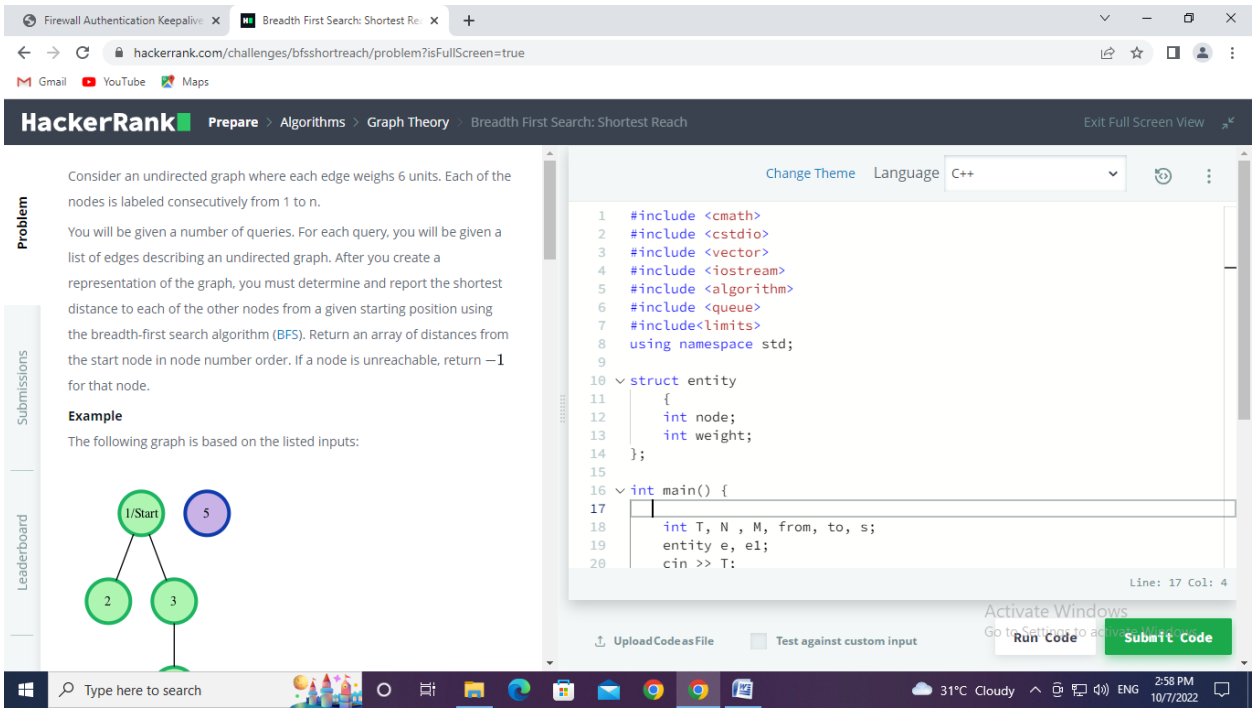
```
#include <cmath>
```

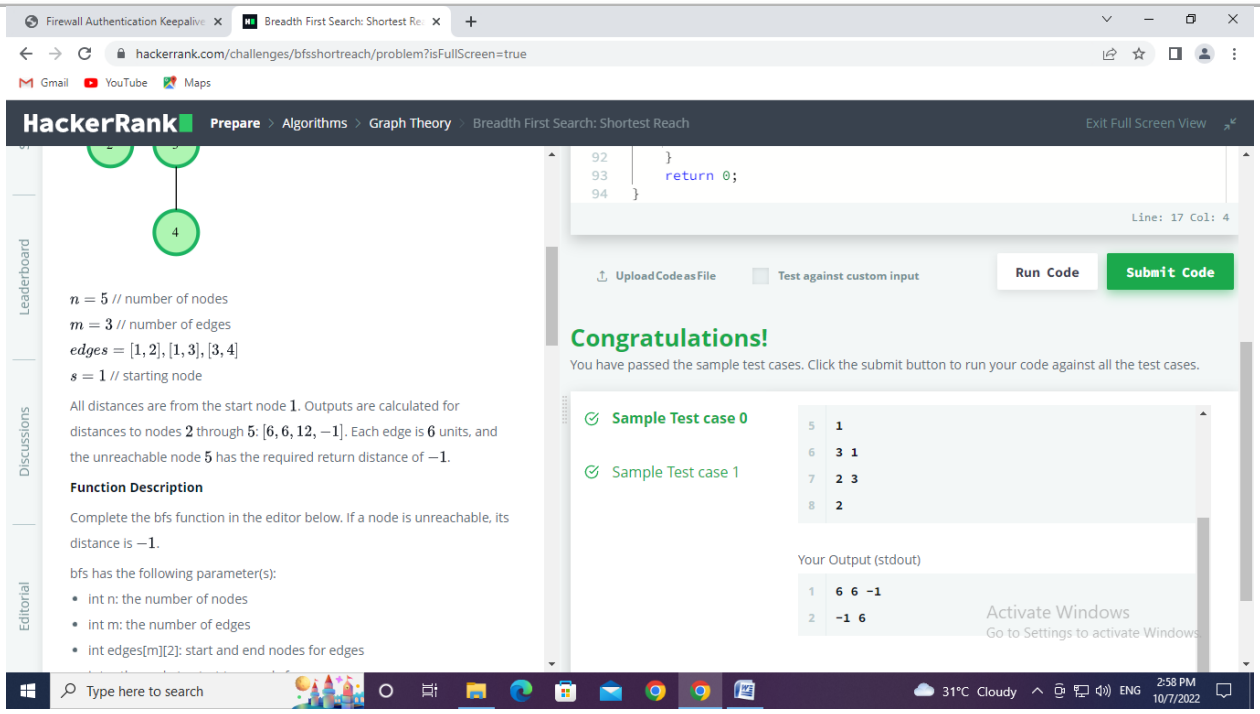
```
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
#include <queue>
#include<limits>
using namespace std;

struct entity
    {
    int node;
    int weight;
};

int main() {

    int T, N , M, from, to, s;
    entity e, e1;
    cin >> T;
    for(int i= 0; i < T; i++)
    {
        cin >> N >> M;
    vector<vector<int>> aList(N);
    vector<int> output(N,numeric_limits<int>::max());
    vector<int> finished(N, -1);
    vector<int>::iterator it;
    for(int i = 0 ; i < M; i++)
        {
            cin >> from >> to;
            it = find (aList[from-1].begin(), aList[from-1].end(), to -1);
            if (it == aList[from-1].end())
            {
                aList[from-1].push_back(to - 1);
            aList[to-1].push_back(from - 1);
            }
    }
    cin >> s;
        output[s-1] = 0;
   //     cout << s << endl;
```

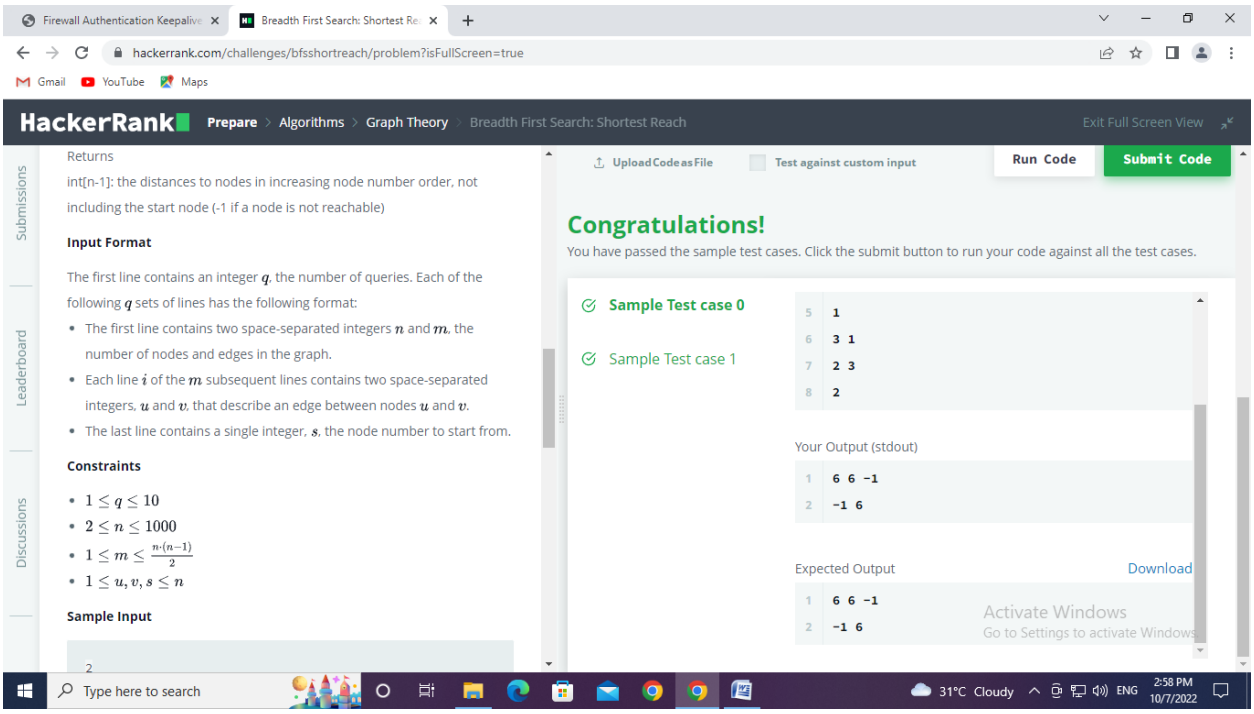
```
  queue<entity> myqueue;
/*      for(int i = 0 ; i < N; i++)
          {
          cout << i << "\t";
          for(int j = 0; j < aList[i].size(); j++)
              {
              cout << aList[i][j] << " ";
          }
          cout << endl;
      }*/
  for(int v : aList[s-1])
      {
     // cout << v;
      e.node = v;
      e.weight = 6;
      myqueue.push(e);

  }
      finished[s-1] = 1;
  while (!myqueue.empty())
{

      e = myqueue.front();
      //cout << e.node << " " << e.weight << endl;
      if (e.weight < output[e.node])
          {
          output[e.node] = e.weight;
      }
      finished[e.node] = 1;
      myqueue.pop();
      for(int v: aList[e.node])
          {
          if(finished[v] != 1)
              {
      e1.node = v;
      e1.weight = 6 + e.weight;
      myqueue.push(e1);
          }
      }
```

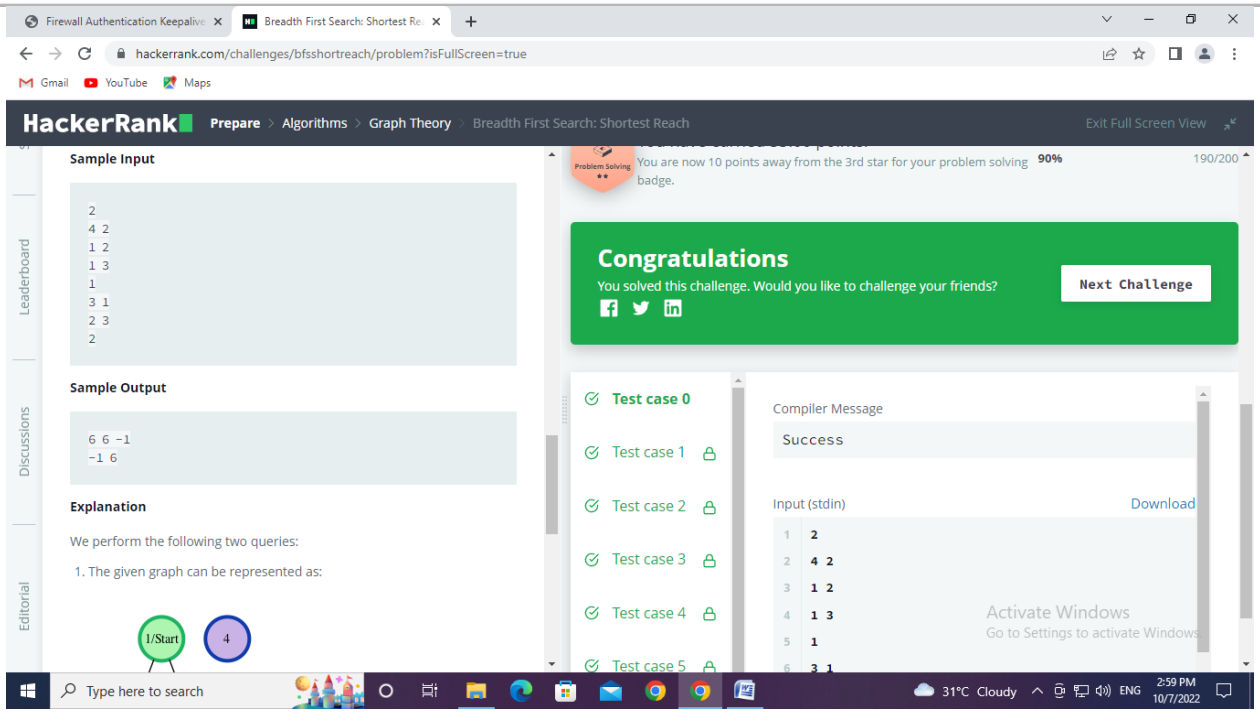
```
        }
        for( int i = 0 ; i < output.size() ;  i++)
            {
            if(i!= s-1)
                {
            if( output[i]!= numeric_limits<int>::max())
                cout << output[i] << " ";
            else
                cout << -1 << " ";
            }
        }
            cout << endl;
        }
        return 0;
    }
```

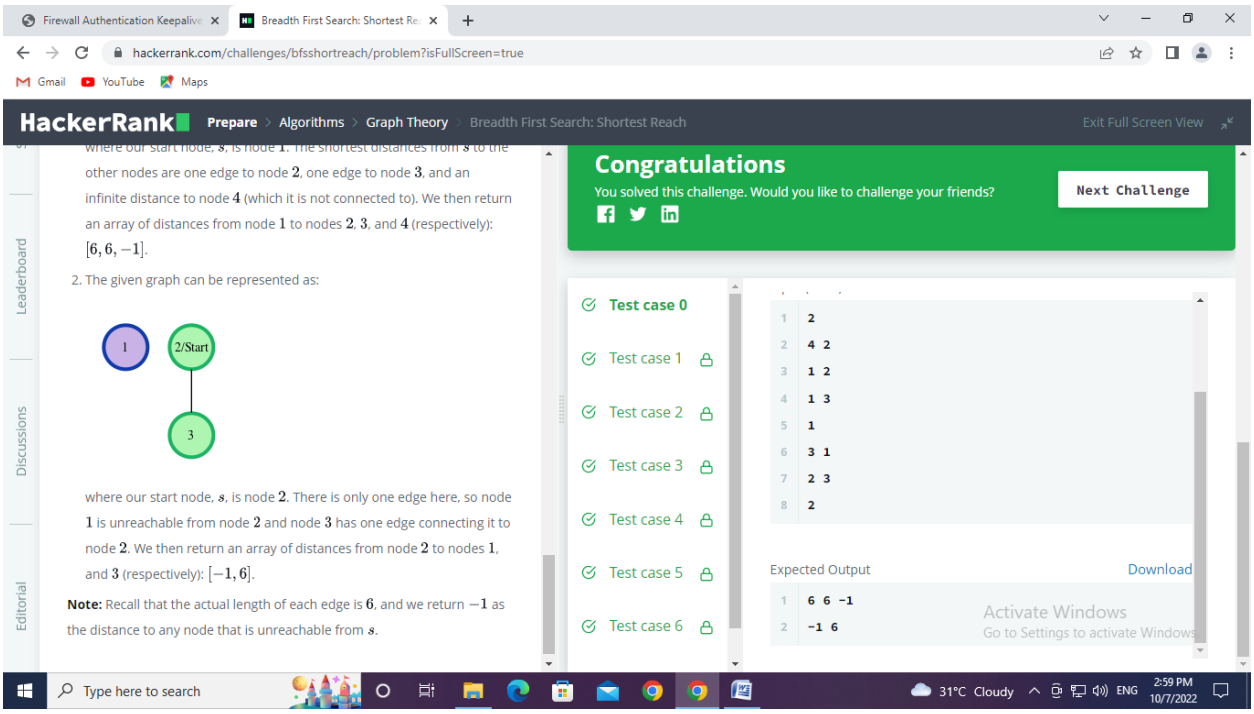
5. Result/Output/Writing Summary:




Firewall Authentication Keepalive
Breadth First Search: Shortest Re
hackerrank.com/challenges/bfsshortreach/problem?isFullScreen=true
Gmail
YouTube
Maps
HackerRank
Prepare
Algorithms
Graph Theory
Breadth First Search: Shortest Reach
Exit Full Screen View
Leaderboard
Discussions
Editorial
4
$n = 5$ // number of nodes
$m = 3$ // number of edges
$edges = [1, 2], [1, 3], [3, 4]$
$s = 1$ // starting node
All distances are from the start node 1. Outputs are calculated for distances to nodes 2 through 5: $[6, 6, 12, -1]$. Each edge is 6 units, and the unreachable node 5 has the required return distance of -1.
Function Description
Complete the bfs function in the editor below. If a node is unreachable, its distance is -1.
bfs has the following parameter(s):
int n: the number of nodes
int m: the number of edges
int edges[m][2]: start and end nodes for edges
92 }
93 return 0;
94 }
Line: 17 Col: 4
Upload Code as File
Test against custom input
Run Code
Submit Code
Congratulations!
You have passed the sample test case. Click the submit button to run your code against all the test cases.
Sample Test case 0
Sample Test case 1
5 1
6 3 1
7 2 3
8 2
Your Output (stdout)
1 6 6 -1
2 -1 6
Activate Windows
Go to Settings to activate Windows.
Type here to search
31°C Cloudy
ENG
2:58 PM
10/7/2022


Firewall Authentication Keepalive
Breadth First Search: Shortest Re
hackerrank.com/challenges/bfsshortreach/problem?isFullScreen=true
Gmail
YouTube
Maps
HackerRank
Prepare
Algorithms
Graph Theory
Breadth First Search: Shortest Reach
Exit Full Screen View
Submissions
Leaderboard
Discussions
Returns
int[n-1]: the distances to nodes in increasing node number order, not including the start node (-1 if a node is not reachable)
Input Format
The first line contains an integer q, the number of queries. Each of the following q sets of lines has the following format:
The first line contains two space-separated integers n and m, the number of nodes and edges in the graph.
Each line i of the m subsequent lines contains two space-separated integers, u and v, that describe an edge between nodes u and v.
The last line contains a single integer, s, the node number to start from.
Constraints
$1 \leq q \leq 10$
$2 \leq n \leq 1000$
$1 \leq m \leq \frac{n \cdot (n-1)}{2}$
$1 \leq u, v, s \leq n$
Sample Input
2
Upload Code as File
Test against custom input
Run Code
Submit Code
Congratulations!
You have passed the sample test case. Click the submit button to run your code against all the test cases.
Sample Test case 0
Sample Test case 1
5 1
6 3 1
7 2 3
8 2
Your Output (stdout)
1 6 6 -1
2 -1 6
Expected Output
Download
1 6 6 -1
2 -1 6
Activate Windows
Go to Settings to activate Windows.
Type here to search
31°C Cloudy
ENG
2:58 PM
10/7/2022


Firewall Authentication Keepalive
Breadth First Search: Shortest Re...
hackerrank.com/challenges/bfsshortreach/problem?isFullScreen=true
Gmail
YouTube
Maps
HackerRank
Prepare
Algorithms
Graph Theory
Breadth First Search: Shortest Reach
Exit Full Screen View
Leaderboard
Discussions
Editorial
Sample Input
2
4 2
1 2
1 3
1
3 1
2 3
2
Sample Output
6 6 -1
-1 6
Explanation
We perform the following two queries:
1. The given graph can be represented as:
1/Start
4
You are now 10 points away from the 3rd star for your problem solving badge.
90%
190/200
Congratulations
You solved this challenge. Would you like to challenge your friends?
Next Challenge
Test case 0
Test case 1
Test case 2
Test case 3
Test case 4
Test case 5
Compiler Message
Success
Input (stdin)
Download
2
4 2
1 2
1 3
1
3 1
Activate Windows
Go to Settings to activate Windows.
Type here to search
31°C Cloudy
ENG
2:59 PM
10/7/2022


Firewall Authentication Keepalive
Breadth First Search: Shortest Re...
hackerrank.com/challenges/bfsshortreach/problem?isFullScreen=true
Gmail
YouTube
Maps
HackerRank
Prepare
Algorithms
Graph Theory
Breadth First Search: Shortest Reach
Exit Full Screen View
Leaderboard
Discussions
Editorial
where our start node, s, is node 1. The shortest distances from s to the other nodes are one edge to node 2, one edge to node 3, and an infinite distance to node 4 (which it is not connected to). We then return an array of distances from node 1 to nodes 2, 3, and 4 (respectively): [6, 6, −1].
2. The given graph can be represented as:
1
2/Start
3
where our start node, s, is node 2. There is only one edge here, so node 1 is unreachable from node 2 and node 3 has one edge connecting it to node 2. We then return an array of distances from node 2 to nodes 1, and 3 (respectively): [−1, 6].
Note: Recall that the actual length of each edge is 6, and we return −1 as the distance to any node that is unreachable from s.
Congratulations
You solved this challenge. Would you like to challenge your friends?
Next Challenge
Test case 0
Test case 1
Test case 2
Test case 3
Test case 4
Test case 5
Test case 6
2
4 2
1 2
1 3
1
3 1
2 3
2
Expected Output
Download
6 6 -1
-1 6
Activate Windows
Go to Settings to activate Windows.
Type here to search
31°C Cloudy
ENG
2:59 PM
10/7/2022

Experiment 5.2

1. **Aim/Overview of the practical:**

To implement the concept of Graphs.

Markov takes out his Snakes and Ladders game, stares at the board and wonders: "If I can always roll the die to whatever number I want, what would be the least number of rolls to reach the destination?"

https://www.hackerrank.com/challenges/the-quickest-way-up/problem?isFullScreen=true

2. **Apparatus / Simulator Used:**
   - Windows 7 or above
   - Google Chrome

3. **Objective:**
   - To understand the concept of graphs.
   - To implement the concept of Graphs.

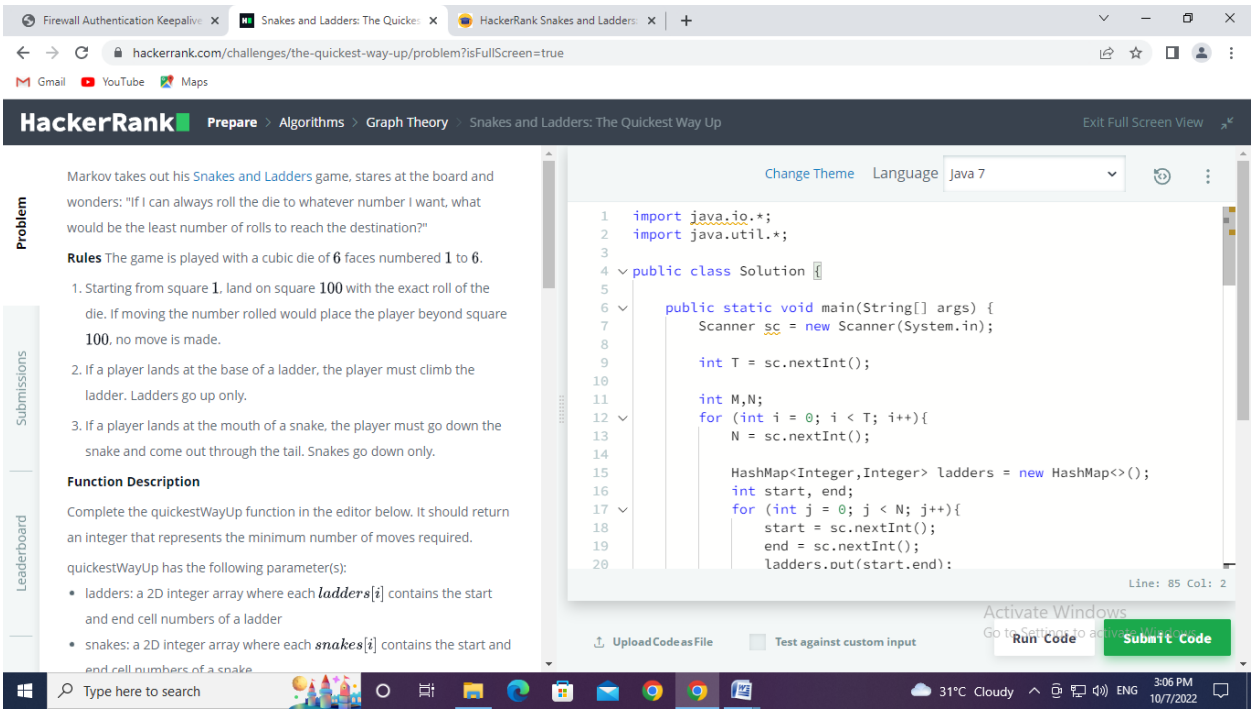
4. **Code:**

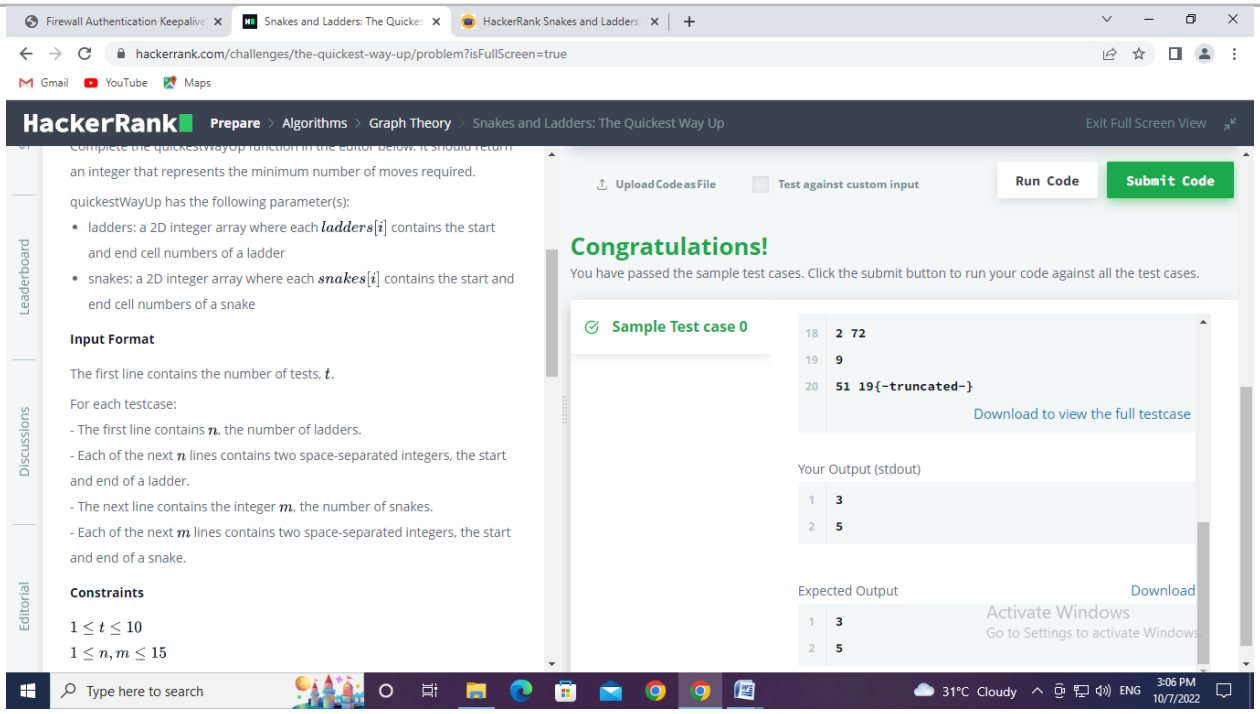
```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int T = sc.nextInt();

        int M,N;
```

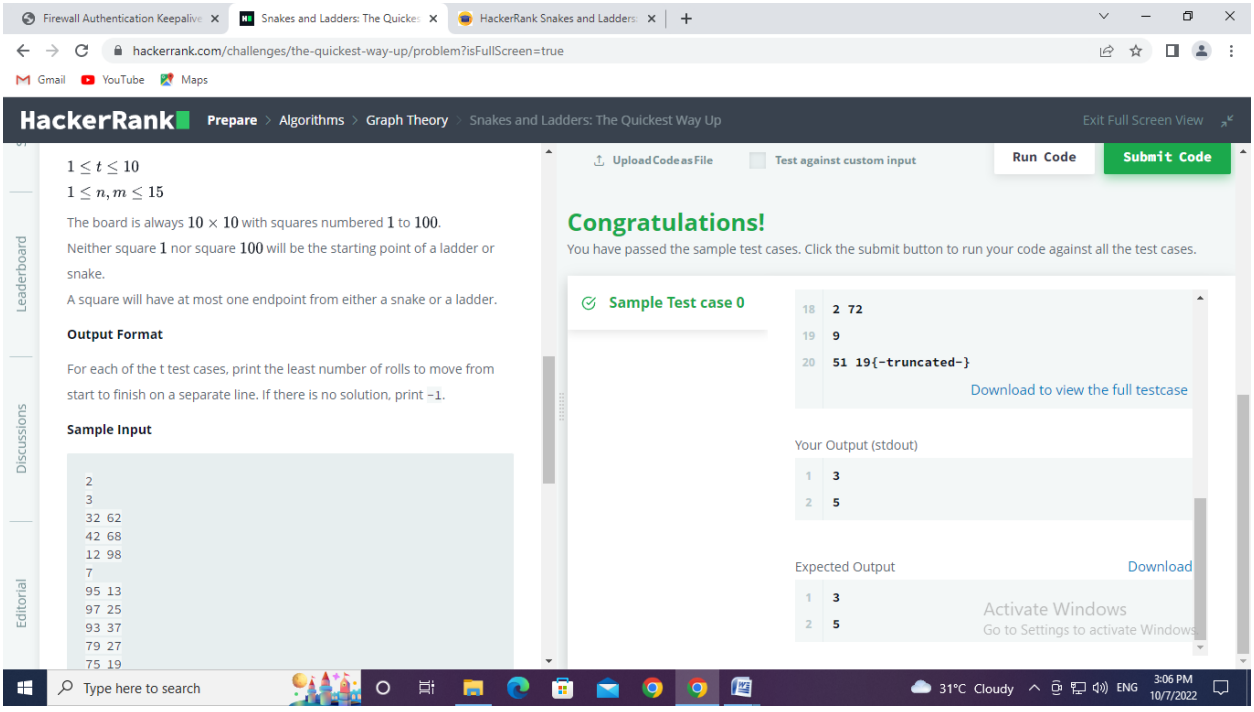
```
        for (int i = 0; i < T; i++){
            N = sc.nextInt();

            HashMap<Integer,Integer> ladders = new HashMap<>();
            int start, end;
            for (int j = 0; j < N; j++){
                start = sc.nextInt();
                end = sc.nextInt();
                ladders.put(start,end);
            }

            HashMap<Integer,Integer> snakes = new HashMap<>();
            M = sc.nextInt();
            for (int j = 0; j < M; j++){
                start = sc.nextInt();
                end = sc.nextInt();
                snakes.put(start, end);
            }

            int[] distances = new int[100];
            for (int j = 0; j < 100; j++){
                distances[j] = Integer.MAX_VALUE;
            }

            getShortestPathToEnd(getGameGraph(ladders, snakes), 1, distances
, 0);

            System.out.println(distances[99] == Integer.MAX_VALUE ? -
1 : distances[99]);
        }
    }

    private static int getShortestPathToEnd(HashMap<Integer,HashSet<Integer>
> graph, int start, int[] distances, int depth){
        if (distances[start-1] > depth){
            distances[start-1] = depth;
        }
        else{
            return 0;
```

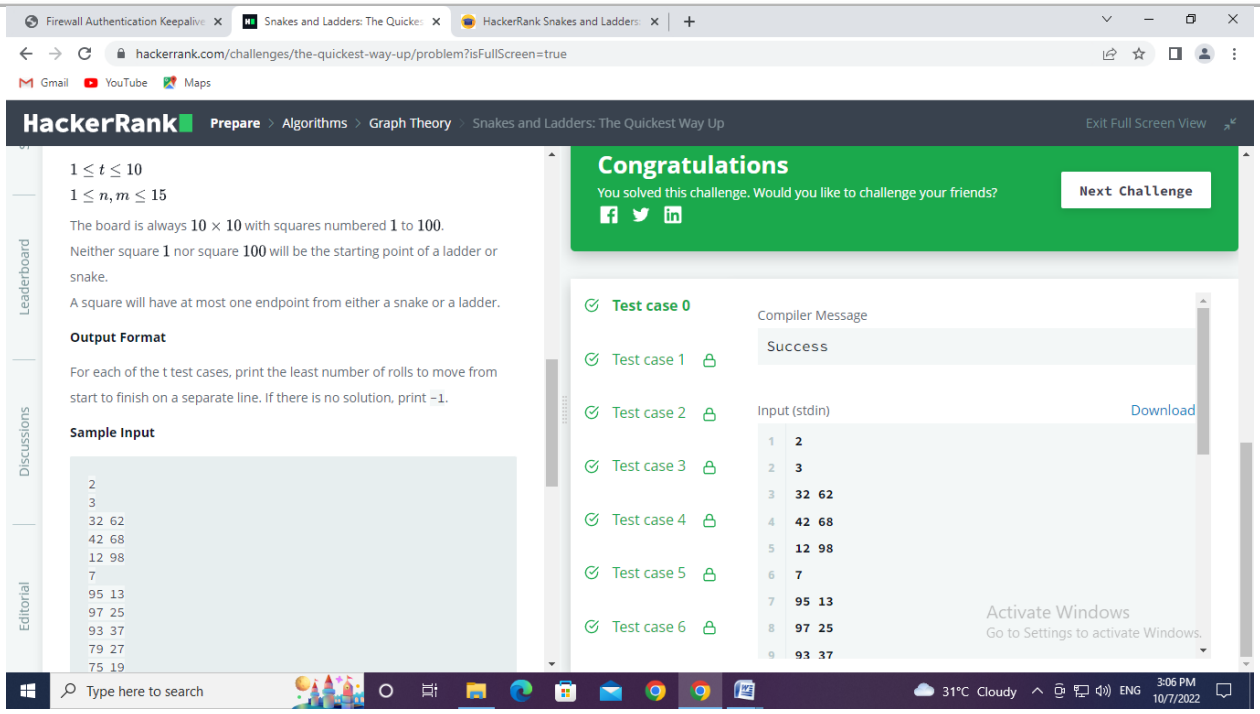
```
        }

        if (!graph.get(start).isEmpty()){
            for (Integer child : graph.get(start)){
                //System.out.println(start + " - " + child);
                getShortestPathToEnd(graph, child, distances, depth + 1);
            }

            return 0;
        }
        else{
            return -1;
        }
    }

    private static HashMap<Integer,HashSet<Integer>> getGameGraph(HashMap<In
teger,Integer> ladders, HashMap<Integer,Integer> snakes){
        HashMap<Integer, HashSet<Integer>> graph = new HashMap<>();

        HashSet<Integer> neighbours;
        for (int i = 1; i <= 100; i++){
            neighbours = new HashSet<Integer>();
            for (int j = 1; j <= 6 && (i + j <= 100); j++){
                if(ladders.containsKey(i+j)){
                    neighbours.add(ladders.get(i+j));
                }
                else if (snakes.containsKey(i+j)){
                    neighbours.add(snakes.get(i+j));
                }
                else{
                    neighbours.add(i+j);
                }
            }
            graph.put(i, neighbours);
        }

        return graph;
    }
}
```

5. Result/Output/Writing Summary:




Firewall Authentication Keepalive
Snakes and Ladders: The Quicke
HackerRank Snakes and Ladders
hackerrank.com/challenges/the-quickest-way-up/problem?isFullScreen=true
Gmail
YouTube
Maps
HackerRank
Prepare
Algorithms
Graph Theory
Snakes and Ladders: The Quickest Way Up
Exit Full Screen View
Leaderboard
Discussions
Editorial
an integer that represents the minimum number of moves required.
quickestWayUp has the following parameter(s):
ladders: a 2D integer array where each ladders[i] contains the start and end cell numbers of a ladder
snakes: a 2D integer array where each snakes[i] contains the start and end cell numbers of a snake
Input Format
The first line contains the number of tests, t.
For each testcase:
- The first line contains n, the number of ladders.
- Each of the next n lines contains two space-separated integers, the start and end of a ladder.
- The next line contains the integer m, the number of snakes.
- Each of the next m lines contains two space-separated integers, the start and end of a snake.
Constraints
1 ≤ t ≤ 10
1 ≤ n, m ≤ 15
Upload Code as File
Test against custom input
Run Code
Submit Code
Congratulations!
You have passed the sample test cases. Click the submit button to run your code against all the test cases.
Sample Test case 0
2 72
9
51 19{-truncated-}
Download to view the full testcase
Your Output (stdout)
3
5
Expected Output
Download
3
5
Activate Windows
Go to Settings to activate Windows.
Type here to search
31°C Cloudy
ENG
3:06 PM
10/7/2022


Firewall Authentication Keepalive
Snakes and Ladders: The Quicke
HackerRank Snakes and Ladders
hackerrank.com/challenges/the-quickest-way-up/problem?isFullScreen=true
Gmail
YouTube
Maps
HackerRank
Prepare
Algorithms
Graph Theory
Snakes and Ladders: The Quickest Way Up
Exit Full Screen View
Leaderboard
Discussions
Editorial
1 ≤ t ≤ 10
1 ≤ n, m ≤ 15
The board is always 10 × 10 with squares numbered 1 to 100.
Neither square 1 nor square 100 will be the starting point of a ladder or snake.
A square will have at most one endpoint from either a snake or a ladder.
Output Format
For each of the t test cases, print the least number of rolls to move from start to finish on a separate line. If there is no solution, print -1.
Sample Input
2
3
32 62
42 68
12 98
7
95 13
97 25
93 37
79 27
75 19
Upload Code as File
Test against custom input
Run Code
Submit Code
Congratulations!
You have passed the sample test cases. Click the submit button to run your code against all the test cases.
Sample Test case 0
2 72
9
51 19{-truncated-}
Download to view the full testcase
Your Output (stdout)
3
5
Expected Output
Download
3
5
Activate Windows
Go to Settings to activate Windows.
Type here to search
31°C Cloudy
ENG
3:06 PM
10/7/2022



$1 \leq t \leq 10$

$1 \leq n, m \leq 15$

The board is always 10×10 with squares numbered 1 to 100.

Neither square 1 nor square 100 will be the starting point of a ladder or snake.

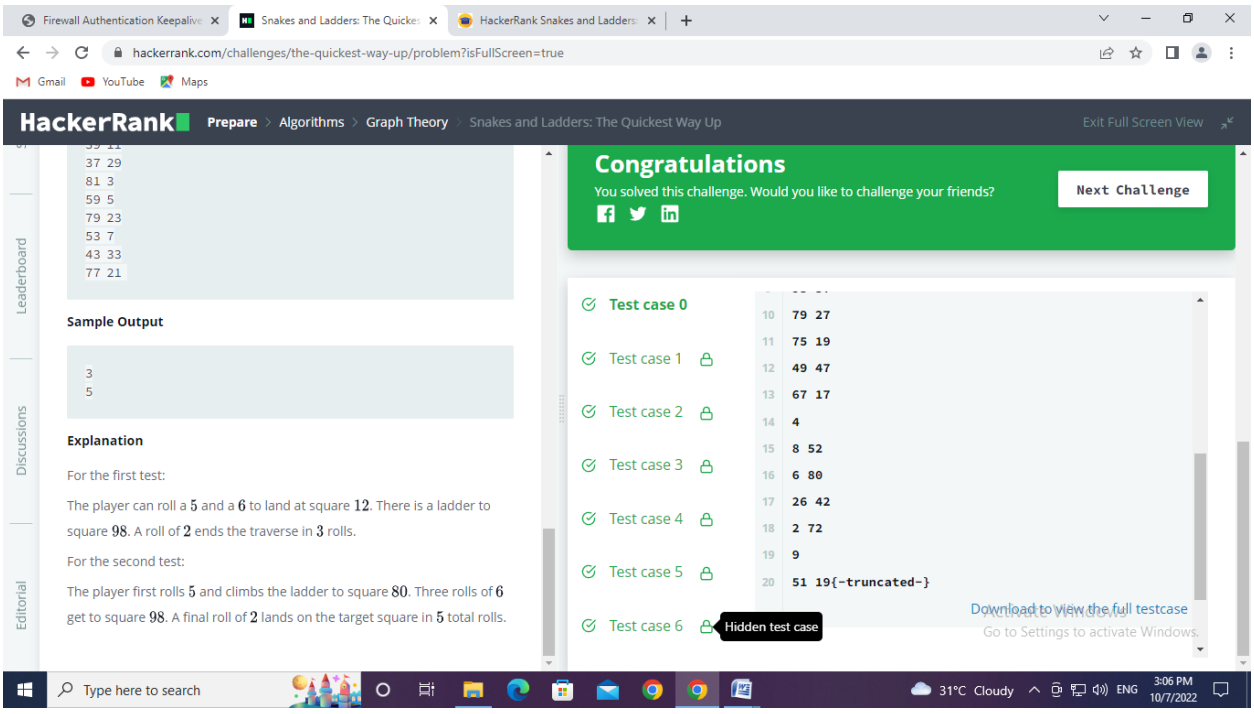
A square will have at most one endpoint from either a snake or a ladder.

Output Format

For each of the t test cases, print the least number of rolls to move from start to finish on a separate line. If there is no solution, print `-1`.

Sample Input

```
2
3
32 62
42 68
12 98
7
95 13
97 25
93 37
79 27
75 19
```



Sample Output

```
3
5
```

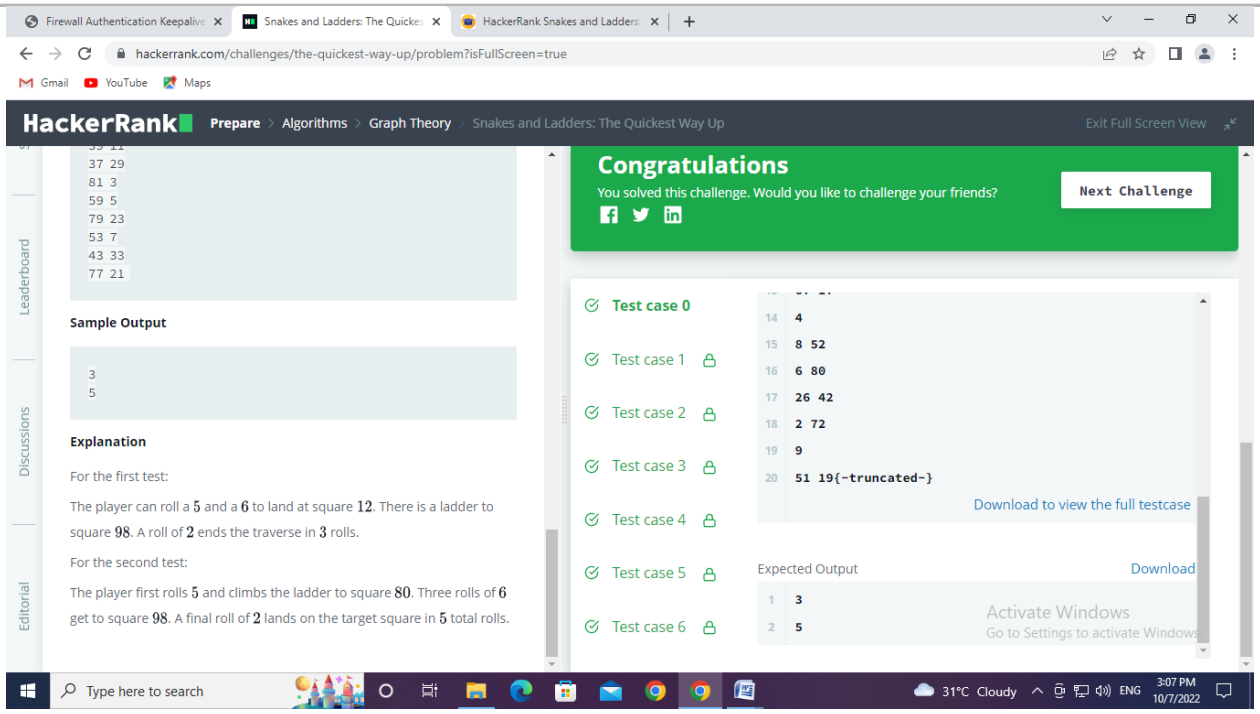
Explanation

For the first test:

The player can roll a 5 and a 6 to land at square 12. There is a ladder to square 98. A roll of 2 ends the traverse in 3 rolls.

For the second test:

The player first rolls 5 and climbs the ladder to square 80. Three rolls of 6 get to square 98. A final roll of 2 lands on the target square in 5 total rolls.



Learning outcomes (What I have learnt):

- Learned about the concept of graphs.
- Learned about implement the concept of Graphs.
- Learned about BFS.
- Learned about the snake and ladder concept using Graph

Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.			
2.			
3.			