

# **TERM 242 ICS-202**

## **PROJECT GUIDELINES**

---

This document provides a guideline for ICS-202 Lab project. The students in the group are required to arrange a team meeting by themselves and distribute the tasks evenly.

### **TOTAL MARKS:**

30 scaled down to 5

### **DEADLINE TO SUBMIT THE TEAM:**

March 16, 2025

### **PROJECT SUBMISSION:**

Students are required to submit the project's product by the one day before the presentation in Week 15.

### **PROJECT PRESENTATION (10-15 MINS):**

- Project Presentation will be held in Week 15 during the regular class time.
- All members from the project group will give the presentation. (Divide the material equally)
- All the members of the group are required to give a live demonstration of the project code.
- Each member of the group will run and explain a specific section of the code.

## PROJECT TASK:

# TREASURE QUEST

---

You are joining in a venture for treasure quest. In a secret chamber which the floor has  $n \times m$  tiles there is a place of treasure that is monitored by  $k$  monsters and you want to get the treasure without being seen by any of the monsters. The  $i$ -th monster is initially located at  $(r_i, c_i)$  and covers distance  $d_i$ , which means monster  $i$  monitors any tile  $(r, c)$  in the chamber as long as  $(r, c)$  is reachable from  $(r_i, c_i)$  in  $d_i$  moves. The chamber has walls which neither you nor the monster can occupy or pass through. The monsters and you may move to any of the 4 adjacent tiles in a single move as long as it is empty.

Given your initial location, find the *shortest path* to the get the treasure without touching tiles monitored by monsters.

### **Input**

Your program will receive input from standard input.

The first line contains three space-separated positive integers  $n$ ,  $m$ , and  $k$ , representing the number of rows of the chamber, the number of columns of the chamber, and the number of monsters, respectively. The following  $n$  lines each contain  $m$  characters representing a map of the chamber. Each character is one of the following:

- : represents an empty tile,
- # : represents a wall
- S : represents your initial location
- E : represents the treasure

There is only one S and E. The border of the chamber is always a wall #.

In the next  $k$  lines, the  $i$ -th line contains three space-separated integers  $r_i$ ,  $c_i$ , and  $d_i$ , representing the initial location and movement range of the  $i$ -th monster. The top-left corner is represented as  $(1, 1)$ , and the bottom-right corner is represented as  $(n, m)$ .

The program must be able to get rid of all possible mistakes in the input.

### **Output**

Your program should write to standard output.

Print exactly one line. The line should contain the length of the shortest path that you can take to move to the get the treasure without touching cells monitored by monsters if it is possible, otherwise print IMPOSSIBLE instead. Note that your initial location or the treasure may already be monitored by a monster, in which case you should print IMPOSSIBLE.

## Constraints

- ✓  $5 \leq n, m \leq 10^3$
- ✓  $0 \leq k \leq 10^4$
- ✓  $1 < r_i < n; 1 < c_i < m; 0 \leq d_i \leq nm$
- ✓ Monsters cannot be placed on a wall
- ✓ The border of the chamber is always a wall
- ✓ No two monsters are placed at the same location

### Sample Input 1 Sample Output 1

10 10 2 ##### #.....# #.....# #...#...E# #.....# #.....# #S.....# #.....## #.....# ##### 6 7 2 4 4 1	15
--	----

### Sample Input 2 Sample Output 2

9 9 2 ##### #.....# #•#•###•# #.....# #•###•E•# #.....#•# #••S##•# #.....# ##### 4 4 1 5 6 0	8
---	---

## Submission Details

- Submit all Java files.
- Use comments in your code
- Cheating and copying are strictly prohibited and evidence of it will result in a zero grade. Make sure you understand the submitted code completely.

- A report containing the following:
  - the data structure used for the layout of the chamber
  - the data structure used for the shortest path algorithm
  - the shortest path algorithm used in this project and its analysis.
- Zip all the items in a folder and named as *YourkfupmID\_LabSec#*
- Consider the points mentioned in the rubrics below:

Questions Answers (Understanding the codes)				
Responses to Questions		Accurate and confident response to questions, including ability to change the code instantly.		7 points
Team Collaboration		Collaborative effort.		3 points
Program's quality				
Performance Indicator		According the following table		20 points
Performance Indicator	Exemplary (100)	Satisfactory (75)	Developing (50)	Unsatisfactory (25)
PI(1): Design capabilities	Design follows the requirement and considerations for space and time efficiency is evident.	Design follows the requirement but no considerations for space and time efficiency is evident.	Some parts are designed by considering the problem/requirement, but not all.	No evidence of data structure and solution design based on the problem/requirement.
PI(2): Implementation	§ Modular Code with proper encapsulation	One of the following issues:	More than one of the following issues:	§ Code not modular with no proper encapsulation
	§ Proper identity (variables) and behavior (methods)	§ Code not modular with no proper encapsulation	§ Code not modular with no proper encapsulation	§ Ill-structured code
	§ Proper use of structured constructs (if, while, for, etc.)	§ Ill-structured code	§ Ill-structured code	§ Not that well indented
	§ Well indented	§ Not that well indented	§ Not that well indented	§ Some vague variable and method names
		§ Some vague variable and method names	§ Some vague variable and method names	
PI(3): Evaluation	Compiles without warnings or errors. Runs for all cases.	Compiles without warnings or errors. Runs for the general cases, Misses at most two special cases.	Compiles without warnings or errors. Runs for general cases. At least three or more different cases do not run as per the given requirements.	May only run for few cases or does not run/compile.