CS 211 Data Structures and Algorithms Lab
Autumn 2020
Examination

Objective	**To help Anu reach her friend's home**
Total marks	**15**
Due date (without penalty)	**29th November (Sunday) 11:59 pm**
Cut-off date (with penalty : 25%)	**30th November (Monday) 11:59 pm**
Penalty for violating naming convention(s)	**5%**

The backstory

After a hectic online semester, Anu wants to go to her friend Tanu's home which is situated in a village. The village is one of the 100 villages in a district - Anu resides in a city of the same district. The villages are numbered from "00" to "99". Anu knows the route from her home to the village "00". Since this is a surprise visit, she cannot ask Tanu how to reach Tanu's home from the village "00". Fortunately, Anu has a map of all the 100 villages and the roads connecting them. Further, she knows the "number" of the village where Tanu's home is situated (the home village). Also, she knows that the home is at the center of the village, where the village roads meet.

Anu drives to the village "00". When she reaches there, she realizes that she forgot the map at her home. She suddenly recalls that you (her another friend and neighbor) may be of help. She calls you and tells you to collect the map from her home and tell her how to reach Tanu's home. You think of sending a Whatsapp image of the map, but realize that Anu does not have internet right now. Since manually exploring the map is very tedious and takes time, you end up with only one option - to write a computer program with the help of your algorithmic arsenal acquired through the DSA course!

Input

Your program should accept an input file as a command-line argument. A typical execution of your program will be ./a.out map.txt. An input file contains the information on the home village and the roads connecting the villages. Every village has at most four gates through which they are connected to other villages by roads. Every village has a square shape and the 100 villages form a square grid of size 10x10. Each village is numbered using two digits - which can be thought as the X and Y coordinates respectively, i.e., the villages are numbered from "00" to "99"; Anu is currently in the center of village "00".

The first line of the input file contains a number which represents the village where Tanu's home is located (the home village). The second line of the file represents the total number of gates connecting villages, which is equal to the total number of roads (a road connects just two villages). Each line in the remaining lines in the file is of the form xy x'y', which represents a road between the villages xy and x'y'. It is guaranteed that there is a route from village "00" to the home village.

Task

Generate an output file named 'directions.txt' where each line is of the form "move <dir>" where <dir> is either north, east, south, or west. The north is in the direction of +ve Y-axis, the south is in the direction of -ve Y-axis, the east is in the direction of +ve X-axis, and the west is in the direction of -ve X-axis.

Anu will read the lines from directions.txt one by one and do accordingly. For example, if Anu is in the village "00", then "move north" directs her to the village north to "00", i.e., the village "01". Similarly "move east" directs her to the village east to "00", i.e., the village "10". It is not guaranteed that there are roads in all the possible directions from a village. So "move <dir>" is not allowed if there is no road from the current village to the village next in the direction <dir> (Anu will hit the wall of the village in this case). Also, crossing the boundary of the grid is not allowed - for example, if Anu is in village "05", then "move west" is not allowed.

The directions given in the directions.txt must eventually lead Anu to Tanu's home through the roads. Please note that there is no requirement of a shortest route. You can apply any algorithm.

Sample inputs and outputs

You are given with a set of three sample inputs and corresponding outputs. For example, map1.txt is a sample input file and a corresponding output file is directions1.txt. For your convenience an image file is also given corresponding to the input file (for example, see map1.eps). A green thick line denotes a gate between two villages through which there is a road. Please note that the image file is only for reference. Also note that there can be multiple correct output files corresponding to multiple valid routes from the village "00" to the home village. So, it is possible that the output file produced by your algorithm for a sample input file is correct even though it is not the same as the given output file.

Evaluation

You get full marks if the output file correctly directs Anu to Tanu's home. If there is a direction in the output file which is not allowed (crossing the boundary of the grid or hitting the wall of a village), or if there is an ill-formatted line, a total of 25% penalty will be applied (it doesn't matter

how many such lines are there) and such lines will be ignored. If Anu does not reach Tanu's home, then the following evaluation scheme will be applied: Assume that the shortest distance (through roads) between village "00" and home village is **d** and the shortest distance (through roads) from the final place of Anu and the home village is **r**, then you receive (d-r)*15/d marks (if this value is -ve, then you receive 0 marks). **Please note that there is no second evaluation for the examination.**

Submission

- The program you submit should output 'directions.txt' when run.
- The main file of your program should be named as <roll no>.<extension>, where roll no. specifies your roll no. and the extension depends on the language you choose (Usage of **C/C++/Python 3/Java** is mandatory for this examination). Ex: 180040001.c. For java programs, please name the program as Java_<rollno>.java
- We will be using gcc/g++ version 6.3, Java version 1.8, Python 3 version 3.6.5 for evaluating your program. If you are using some other version of gcc or java, mostly your program will run fine while doing the evaluation. Please do not use Python 2.
- Test well before submission. You may use the attached sample input file(s) for testing. The corresponding output file(s) is also attached. Please note that this is only for reference - an error in these files is not a valid reason for an error in your program! We have some hidden inputs with us to test your program. The mark you obtain is purely based on whether your program correctly gives outputs for the hidden inputs.
- If your program has only a single source file, please submit the file as it is. If your program has multiple source files, please submit your code as a zip file where the name of the zip file should be your roll number. It is important that you follow the input/output conventions exactly (including the naming scheme) as we may be doing an automated evaluation. There will be a penalty of 5% (on the mark you deserve otherwise) if you do not follow the naming conventions exactly.
- Follow some coding style uniformly. Provide proper comments in your code.
- Submit only through moodle. Submit well in advance. Any hiccups in the moodle/internet at the last minute is never acceptable as an excuse for late submission. Submissions through email or any other means will be ignored.
- Acknowledge the people (other than the instructor and TA) who helped you to solve this assignment. The details of the help you received and the names of the people who helped you (including internet sources, if applicable) should come in the beginning of the main file as a comment. Copying others' programs and allowing others to copy your program are serious offences and a deserving penalty will be imposed if found.