Digital Pakistan Speed Programing Competition Online Qualifier Round

Instructions

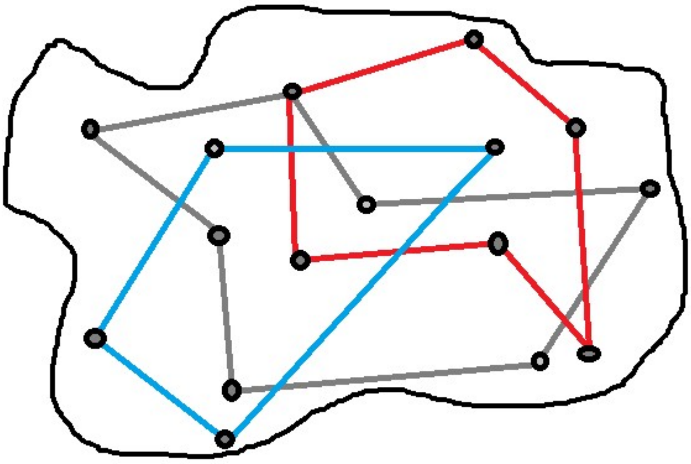
- Do not open the booklet unless you are explicitly told to do so. You can only read these instructions below.
- If you have any question regarding the problems, seek a clarification from the judges using DOMJudge.
- Before submitting a run, make sure that it is executable via command line. For Java, it must be executable via "javac" and for GNU C++ via "g++". Java programmers need to remove any "package" statements and source code's file name must be the same as of main class. C++ programmers need to remove any getch() / system("pause") like statements.
- Do not attach input files while submitting a run, only submit/attach source code files, i.e., *.java or *.cpp or *.py.
- Language supported: C/C++, Java and Python3
- Source code file name should not contain white space or special characters.
- You must take input from Console i.e.: Standard Input Stream (stdin in C, cin in C++, System.in in Java, stdin in Python)
- You must print your output to Console i.e.: Standard Output Stream (stdout in C, cout in C++, System.out in Java)
- Please, don't create/open any file for input or output.
- Please strictly meet the output format requirements as described in problem statements, because your program will be auto judged by computer. Your output will be compared with judge's output byte-by-byte and not tolerate even a difference of single byte. So, be aware! **Pay special attention to spaces, commas, dots, newlines, decimal places, case sensitivity etc.**
- All your programs must meet the time constraint specified.
- The decision of judges will be absolutely final.

Problem 06: Hop On – Hop Off

Time limit: 1 seconds

The local government of a world-renowned city is **improving transport infrastructure** for its tourists. They have launched a huge network of trams, buses, railway lines, and subways that are spread across all corners of the city. It is also interesting to note that all the routes are circular in nature. That means if a tourist hops on a train at one station, he/she can hop off at the same station as the train route completes.

The city is full of cultural, academic, arts, and natural beauty spots, which are a great attraction to tourists, and it takes one complete day to fully enjoy in one place. Since the duration of visitors' stay in the city is only one day, therefore, city administration always suggests them to pin-point their area of interest and select the ways of commuting way ahead of their arrival. In this way, tourists will know which transport they have to hop-on and where to hop-off to pick up the next connecting route to reach their destination with the minimum number of transport changes. Despite all the efforts by the city administration, sometimes it is possible that a visitor may not be able to reach his/her destination as there might not be a possible connection from his/her place of stay.



Three different routes for moving in the city

However, the government observed that not all tourists are well versed with the use of technology; therefore, they are requiring your services to write a code that can help tourists in finding the minimum hops (transport changes) to reach a destination from their place of stay.

Example

Given the following routes, with source and destination

Route1 = 1, 5, 7, 9
Route2 = 2, 5, 9, 8, 16
Route3 = 4, 7, 15, 90
Source = 1, Destination = 90

The solution will be such that the tourist will have the following path 1 5 7 15 90, starting from the source=1 to reach the destination=90, and requiring two hops.

Input Format

The first line of the test case represents ***n*** modes of transport $(1 \leq n \leq 10^3)$. The subsequent ***n*** lines contain space-separated stops for the ***i***th route$(1 \leq i \leq n)$. The subsequent line contains the source and destination, as space-separated values.

Output Format

The output consists of a single value denoting the minimum number of hops required to make it to the destination from the start. If no hop is required, print 0. If no solution exists, print -1.

Sample input	Sample Output

3
1 5 7 9
2 5 9 80 16
4 7 15 900
1 900
	2

2
9 5 3
2 3 1
5 2
	2

3
1 5 7 9
2 5 9 8 16
4 10 15 90
5 90
	-1