CST 370 – Homework 2

Due: **2:00pm on February 21, 2019**

Final Submission Deadline: **2:00pm on February 28, 2019**

About

This homework assignment asks you to apply graph serach and topological sort algorithms to a variety of problems.

There are 8 problems in this homework **but you don't need to do all of them!** You must complete 40 points worth of problems. You can do this by doing:

- Two 15-point problems and one 10-point problem
- All four 10-point problems
- Both 20-point problems
- One 20-point problem and two 10-point problems

The point values correspond to difficulty so the choice of which problems to complete is up to you.

Submissions

Each problem on this assignment be submitted on HackerRank, you must join the homework 2 contest at https://www.hackerrank.com/cst370-s19-hw2 in order to submit.

For each problem, you may submit as many times as you would like, I will only look at the final submission before the due date. You must submit something that shows you've put in the effort before the due date in order to resubmit before the final submission deadline (an empty function does not show effort).

After the due date, you may resubmit as many times as you'd like before the final submission deadline. I will only look at the final resubmission before the deadline. Resubmissions will be assessed a 10% deduction.

Scoring

Your score on HackerRank or Kattis will give you *an approximation* of what your grade will be; however, I will read your code and determine your final grade myself, as detailed on the syllabus.

Problems

1	Euro(ish) Trip	15 points
2	Alien Alphabet	20 points
3	California Road Trip	10 points
4	Advanced Catan	20 points
5	Train Routes	10 points
6	CSUMB Campus Walk	10 points
7	Species Survival	15 points
8	Flight Cancellations	10 points

1. Euro(ish) Trip (15 points)

https://www.hackerrank.com/contests/cst370-s19-hw2/challenges/hw2-euroish

You are planning a vacation around Europe (and some other places because you had a hard time deciding) and wish to determine the number of possible itineraries before booking the trip. However, there are a lot of constraints due to flights, money, picky friends, and hotels. You will receive a directed acyclic graph of the cities you wish to visit, and must produce all possible orders to visit them in alphabetically.

Input

The input describes a graph where an edge from a to b means city a is adjacent to city b.

- The first line contains an integer, n, the number of cities you want to visit.
- The next n lines each contain a city. This makes up an array of verticies.
- The next line contains two space-separated integers: n e
- The next e lines each contain two space-separated integers a and b describing an edge from the vertex indexed at a to the vertex indexed at b.

If you look at the sample input, london, istanbul, paris, and la are adjacent to nyc; london, lisbon, and honolulu are adjacent to la; lisbon and istanbul are adjacent to honolulu; and so on.

Constraints

You can assume the input is valid (i.e., that it represents a directed acyclic graph).

Output

Comma-separated lines representing all possible topological sorts of the input, in alphabetical order. See the hackerrank problem for sample input and output.

This page is intentionally left blank.

2. Alien Alphabet (20 points)

https://www.hackerrank.com/contests/cst370-s19-hw2/challenges/hw2-aliens

You will receive a "sorted" array of words, representing an alphabetized dictionary in an alien language. "Sorted" is in quotations because this array will not appear sorted to the untrained eye - that's because in this alien language, the letter order is different than from English. Your job is to determine the order of characters in the alien language.

For example, given the "sorted" array

```
["caa", "aaa", "aab"]
```

we can determine the alien alphabet must be ordered

```
['c', 'a', 'b']
```

Input

- The first line of input will indicate the number of words n in the alien dictionary.
- The next n lines will be the words, alphabetized.

Constraints

You can assume the alien alphabet consists of 26 or fewer letters, each of which is also in the English alphabet.

Output

A single line with the space-separated alien alphabet in order (no terminating white space). See the hackerrank problem for sample input and output.

This page is intentionally left blank.

3. California Road Trip (10 points)

https://www.hackerrank.com/contests/cst370-s19-hw2/challenges/hw2-roadtrip

You're going on a road trip around California, but you have a limited amount of time. Given a graph representing possible routes you could take, use a graph traversal to figure out the possible places you might end up given that you only have time to make k or fewer stops.

Input

- The first line contains an integer, n, the number of locations you have time to visit in addition to your starting location.
- The next line contains your starting location.
- The remaining input describes a graph where each edge means two locations are adjacent:
  - The next line contains an integer n, the number of locations you're interested in visiting.
  - The next n lines each contain one location. This makes up an array of verticies.
  - The next line contains two space-separated integers: n e
  - The next e lines each contain two space-separated integers a and b describing an edge from the vertex indexed at a to the vertex indexed at b.

Constraints

You may assume the input represents an undirected, connected, cyclic graph.

Output

An alphabetized list of the possible locations you could end up if you stop at n or fewer locations after your starting point, one location per line. See the hackerrank problem for sample input and output.

This page is intentionally left blank.

4. Advanced Catan (20 points)

https://www.hackerrank.com/contests/cst370-s19-hw2/challenges/hw2-advancedcatan

In the Settlers of Catan, you can't place a settlement within two spaces of another settlement. In this advanced version of Catan, the rules are a little different. You can't place a settlement in a location that is within k spaces of all existing settlements (or on top of an existing settlement). (So a settlement that is within k spaces of some existing settlements but not every existing settlement is legal.) Your task is, given a board and a number k, determine the number of offlimit locations for a settlement.

Input

- The first line contains an integer k, distance required from settlements.
- The next line contains an integer x, the number of settlements.
- The next x lines contain the unique identifiers for those k settlements.
- The following line contains an integer n, the number of positions on the board.
- The next n lines contain a unique identifier for that position, followed by a colon, followed by a comma-separated list of adjacent positions.

Constraints

You may assume the positions form an undirected, connected graph.

Output

A single line containing an integer indicating the number of invalid settlement locations. See the hackerrank problem for sample input and output.

This page is intentionally left blank.

5. Train Routes (10 points)

https://www.hackerrank.com/contests/cst370-s19-hw2/challenges/hw2-trains

You're trying to plan a train journey but unfortunately you lost the routes pamphlet. Thankfully, you still have the map, from which you can figure out the stops on each route. You'll receive a graph representing the stops and the ones they're adjacent to, and you need to figure out how many routes it represents. A route consists of all stops that are connected to each other by some path; therefore, distinct routes are not connected to each other.

Input

The input describes a graph where an edge from a to b means stop a is adjacent to stop b.

- The first line contains an integer, n, the number of stops.
- The next n lines each contain a stop. This makes up an array of verticies.
- The next line contains two space-separated integers: n e
- The next e lines each contain two space-separated integers a and b describing an edge from the vertex indexed at a to the vertex indexed at b.

Constraints

You can assume the graph is undirected.

Output

A single line with the number of routes represented. See the hackerrank problem for sample input and output.

This page is intentionally left blank.

6. CSUMB Campus Walk (10 points)

https://www.hackerrank.com/contests/cst370-s19-hw2/challenges/hw2-campuswalk

You need to start doing your CST370 homework, but you don't really feel like it. You decide to take the longest route from your current location on campus to where you plan to do your homework to procrastinate a bit. You're given a graph representing CSUMB's campus, and your goal is to find the path with the most stops to your homework destination without visiting any location more than once.

Input

- The first line contains your current location.
- The second line contains the location where you'll do your homework.
- The remaining input describes a graph where each edge means two locations are adjacent:
  - The next line contains an integer n, the number of locations on campus.
  - The next n lines each contain one location. This makes up an array of verticies.
  - The next line contains two space-separated integers: n e
  - The next e lines each contain two space-separated integers a and b describing an edge from the vertex indexed at a to the vertex indexed at b.

Constraints

You can assume the campus locations form a connected, undirected graph.

Output

The length of the longest possible path you can take your current location to your homework destination on a single line. See the hackerrank problem for sample input and output.

This page is intentionally left blank.

7. Species Survival (15 points)

https://www.hackerrank.com/contests/cst370-s19-hw2/challenges/hw2-species

The wolf is in trouble from years of hunting. One strategy used in endgangered species survival is to move individuals from one location to another so they mix gene pools and are less susceptible to genetic diseases. We have a plan to move wolves from some locations to others, but we'd like to make the maximum moves possible to have the greatest possible diversity in our gene pools. We need to determine how many additional moves we can make without creating a cycle (or we might move back some of the individuals we moved out, undoing our work!). Similarly, we don't want to create any back-edges (undirected edges, or edges A-¿B when we already have B-¿A) so we don't risk moving any of the same animals back. Given the move plan, determine how many more moves we can make within these constraints.

Input

The input describes a graph where an edge from a to b means we'll distribute wolves from a to b.

- The first line contains an integer, n, the number of locations in the move plan.
- The next n lines each contain a location in the plan. This makes up an array of verticies.
- The next line contains two space-separated integers: n e
- The next e lines each contain two space-separated integers a and b describing an edge from the vertex indexed at a to the vertex indexed at b.

Constraints

You can assume the move plan is valid as-is (i.e., contains no cycles and no undirected edges).

Output

A single line containing the number of additional moves we can make. See the hackerrank problem for sample input and output.

This page is intentionally left blank.

8. Flight Cancellations (10 points)

https://www.hackerrank.com/contests/cst370-s19-hw2/challenges/hw2-flights

Airlines haven't been doing so well recently. In addition to make the seats miniscule, getting rid of the individual TVs, no longer serving peanuts, killing pets, and not seating people for wearing clothes they don't like, they're also cancelling flights. They want to cancel all flights between cities A and B where the fewest passengers travel directly from A to B. However, they don't want to remove flights that would mean two cities are no longer connected by air travel (they don't care if this suddenly means you now have to take 4 flights instead of 1). Given origin and destination cities, your job is to figure out if cancelling all flights between those two cities will disconnect two previously connected cities.

Input

- The first two lines contain the cities the airline wishes to cancel flights between.
- The remaining input describes a graph where each edge is a flight between two cities:
  - The next line contains an integer n, the number of cities the airline serves.
  - The next n lines each contain a city served by the airline. This makes up an array of verticies.
  - The next line contains two space-separated integers: n e
  - The next e lines each contain two space-separated integers a and b that describes an edge from the vertex indexed at a to the vertex indexed at b.

Constraints

You can assume the flight graph is originally connected and is undirected - i.e., if there are flights from A-¿B there are flights from B-¿A and a single edge represents both.

Output

A single line containing "True" or "False" indicating whether they can remove the flights. See the hackerrank problem for sample input and output.

This page is intentionally left blank.