Dear candidate,

Following to your application, here’s the next step in our hiring process for Digital Lab. You will continue working on the code as a part of the live interview stages.

Below is a programming problem. Please read all three descriptions thoroughly then create a program to solve the problem. Please upload the solution to github and share the repo with us. If you submit more than one solution, we will review only one. For the solution, we request that you use NodeJS, Java or Go.

You may not use any external libraries to solve this problem, but you may use external libraries or tools for building or testing purposes. Specifically, you may use unit-testing libraries or build tools available for your chosen language.

Please commit all your code into a public GitHub repository and share the url with us (send an email to nikolay\_ryabkov@mckinsey.com)

Please include a brief explanation of your design and assumptions, along with your code, as well as detailed instructions to run your application.

We assess a number of things including the design aspect of your solution and your programming skills. While this is a small problem, we expect you to submit what you believe is production-quality code; code that you’d be able to run, maintain, and evolve. You don’t need to gold-plate your solution, however we are looking for something more than a bare-bones algorithm.

We want our hiring process to be fair, and for everyone to start from the same place. To enable this, we request that you do not share or publish these problems.

Please note that completing this task will require around 6 hours for a person fully familiar with technologies. However, as a general rule, we allow one day from the date that you receive these instructions to submit your code, but you may request more time if needed.  If you have any questions about the code as it relates to your interview process, please let us know.

We wish you luck and look forward to receiving your response.

If you have any questions, please send an email nikolay\_ryabkov@mckinsey.com

Introduction to the problem

•               There must be a way to supply the application with the input data via command line or text file

•               The application must run

•               You should provide sufficient evidence that your solution is complete by, as a minimum, indicating that it works correctly against the supplied test data

•               Bonus points if you also have a working code for deployment in AWS EKS via Terraform.

TRAINS and TOWNS

The local commuter railroad services a number of towns.  Because of monetary concerns, all of the tracks are 'one-way.'  That is, a route from A to B does not imply the existence of a route from B to A. In fact, even if both of these routes do happen to exist, they are distinct and are not necessarily the same distance!

The purpose of this problem is to help the railroad provide its customers with information about the routes.  In particular, you will compute the distance along a certain route, the number of different routes between two towns, and the shortest route between two towns.

Input:  A directed graph where a node represents a town and an edge represents a route between two towns.  The weighting of the edge represents the distance between the two towns.  A given route will never appear more than once, and for a given route, the starting and ending town will not be the same town.

Output: For test input 1 through 5, if no such route exists, output 'NO SUCH ROUTE'.  Otherwise, follow the route as given; do not make any extra stops!  For example, the first problem means to start at city A, then travel directly to city B (a distance of 5), then directly to city C (a distance of 4).

1. The distance of the route A-B-C.

2. The distance of the route A-D.

3. The distance of the route A-D-C.

4. The distance of the route A-E-B-C-D.

5. The distance of the route A-E-D.

6. The number of trips starting at C and ending at C with a maximum of 3 stops.  In the sample data below, there are two such trips: C-D-C (2 stops). and C-E-B-C (3 stops).

7. The number of trips starting at A and ending at C with exactly 4 stops.  In the sample data below, there are three such trips: A to C (via B,C,D); A to C (via D,C,D); and A to C (via D,E,B).

8. The length of the shortest route (in terms of distance to travel) from A to C.

9. The length of the shortest route (in terms of distance to travel) from B to B.

10.The number of different routes from C to C with a distance of less than 30.  In the sample data, the trips are: CDC, CEBC, CEBCDC, CDCEBC, CDEBC, CEBCEBC, CEBCEBCEBC.

Test Input:

For the test input, the towns are named using the first few letters of the alphabet from A to D.  A route between two towns (A to B) with a distance of 5 is represented as AB5.

Graph: AB5, BC4, CD8, DC8, DE6, AD5, CE2, EB3, AE7

Expected Output:

Output #1: 9

Output #2: 5

Output #3: 13

Output #4: 22

Output #5: NO SUCH ROUTE

Output #6: 2

Output #7: 3

Output #8: 9

Output #9: 9

Output #10: 7