A

CSE221: Algorithms (Lab)

Semester: **Summer 2023** Examination: **Lab Final**

Duration: **75 minutes** Full marks: **20**

Part 1

You're managing an expedited courier service in a bustling city. Your couriers need to navigate through a network of ***N*** junctions connected by ***M*** streets to deliver packages as quickly as possible. Each street has a time value representing the time it takes to traverse it. Your goal is to develop an algorithm that helps your couriers find the quickest route to their destinations.

Part 2

After a few years of being in service, you invest in adding **1** new vehicle for even faster delivery. You decide that this new vehicle should operate on **the path with the most intermediate nodes** (not necessarily the most time consuming), effectively **cutting the time of that path in half**. Make adequate modifications to the solution to Part 1 to accommodate this change.

Input

- You should use the **same input file for both parts**.
- The first line of the input contains two integers, ***N*** and ***M*** **(1 <= *N* <= 1000, 1 <= *M* <= 100000)** denoting the number of junctions and streets in the city, respectively.
- The next ***M*** lines each contain three integers, ***u***, ***v*** **(1 <= *u*, *v* <= *N*)**, and ***t*** **(1 <= *t* <= 100)** denoting a street from location ***u*** to location ***v*** with time ***t***.
- The last line of the input contains an integer ***S*** **(1 <= *S* <= *N*)** denoting the source node.

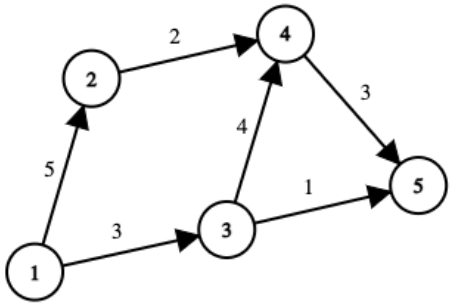
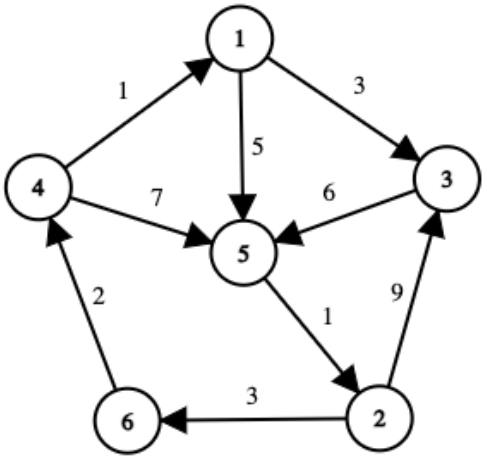
Output

- Output an adjacency list for the given street-network.
- For each part:
  - Output ***N*** space-separated integers, where the ***i-th*** integer represents the shortest time from the source to location ***i***.
  - If a place is not reachable from the source, output **-1** instead.
- You should output to the **same file for both parts**.

Marks breakdown

- Input from file ----------> **2**
- Display adjacency list -------> **4**
- Implement Part 1 ----------> **7**
- Implement Part 2 ----------> **5**
- Output to file ----------> **2**
- **Total** ----------> **20**

(continued on next page)

Sample Input 1: 5 6 1 2 5 1 3 3 2 4 2 3 4 4 3 5 1 4 5 3 1	**Sample Output 1:** Adjacency List: 1 : (2, 5) (3, 3) 2 : (4, 2) 3 : (4, 4) (5, 1) 4 : (5, 3) 5 : Before adding vehicle: 0 5 3 7 4 After adding vehicle: 0 5 3 3 4	
Sample Input 2: 6 10 1 3 3 1 5 5 2 2 4 2 3 9 3 5 6 4 5 7 6 4 2 5 2 1 2 6 3 4 1 1 5	**Sample Output 2:** Adjacency List: 1 : (3, 3) (5, 5) 2 : (2, 4) (3, 9) (6, 3) 3 : (5, 6) 4 : (5, 7) (1, 1) 5 : (2, 1) 6 : (4, 2) Before adding vehicle: 7 1 10 6 0 4 After adding vehicle: 3 1 10 6 0 4	

End