Question 1:

A city has **N** districts numbered from **0** to **N-1**. These districts are connected by **M** two-way roads. No two districts have more than one direct road between them.

The central district of the city is always numbered **0**. The city administration wants to ensure that all other districts can reach the central district through the road system, either directly or indirectly. However, some districts may be isolated due to a lack of connectivity to the center.

Write a program to identify the list of districts that cannot reach the central district. If all districts in the city are connected to the central district, print: **"Connected City"**.

The input: are stored in the ***input.txt*** text file:

- The first line contains two integers **N** and **M**, representing the number of districts and the number of roads, respectively.
- The next **M** lines each contain two integers **u** and **v**, representing a two-way road between district **u** and district **v**.

The output: the results need to be saved to the ***output.txt*** text file:

- If there are districts that cannot reach the central district, print the list of isolated districts (sorted in ascending order and separated by commas ,).
- If all districts can reach the central district, print: **"Connected City"**.

Sample Input 1	*Sample Output 1*
6 4 0 1 1 2 3 4 4 5	3,4,5

Sample Input 2	*Sample Output 2*
5 4 0 1 1 2 2 3 3 4	Connected City

Sample Input 3	*Sample Output 3*
7 3 0 1 2 3 4 5	2,3,4,5,6

Question 2:

A port has **N** docks numbered from **0** to **N-1**. Some docks have **direct transportation routes** connecting them, allowing ships to travel between them. Dock **0** is the **coordination center**, responsible for dispatching ships and managing logistics.

The port management wants to ensure that **every dock can receive dispatch orders** from the coordination center, either **directly** or **indirectly** through connected docks. However, some docks might be **isolated** and unable to receive orders.

Your task is to determine which docks **cannot** receive dispatch orders from the coordination center.

The input: are stored in the ***input.txt*** text file:

- The first line contains **two integers**:
  - **N**: The number of docks.
  - **M**: The number of transportation routes between docks.
- The next **M** lines each contain **two integers A and B**, indicating a transportation route between docks **A** and **B**.

The output: the results need to be saved to the ***output.txt*** text file:

- A **list of integers**: The dock numbers that **cannot** receive orders from the coordination center (dock **0**). The numbers should be **comma-separated**.
- If **all docks** can receive orders, print: **"Operational Dispatch System"**.

Sample Input 1	*Sample Output 1*
7 5 0 1 1 2 3 4 4 5 5 6	3,4,5,6

Sample Input 2	*Sample Output 2*
6 5 0 1 1 2 2 3 3 4 4 5	Operational Dispatch System

Question 3:

A city consists of **N** areas numbered from **0** to **N-1**. The city has a **water supply system** with a **main water station** located in area **0**. Some areas are directly connected by **water pipelines**, allowing water to flow from one area to another.

The city administration wants to ensure that **every area in the city receives water**. However, due to incomplete pipeline connections, some areas may be **isolated** and unable to receive water from the main station.

Your task is to identify the areas that **do not receive water** from the main station. If all areas are connected to the main station (either **directly or indirectly**), print **"Stable Water Supply"**.

The input: are stored in the ***input.txt*** text file:

- The **first line** contains **two integers**:
  - **N** → Number of areas.
  - M → Number of pipelines connecting areas.
- The next **M** lines each contain **two integers A and B**, representing a **pipeline between areas A and B**.

The output: the results need to be saved to the ***output.txt*** text file:

- A **comma-separated list** of the area numbers that **cannot receive water** from area **0**.
- If **all areas** receive water, print **"Stable Water Supply"**

Sample Input 1	*Sample Output 1*
6 4 0 1 1 2 3 4 4 5	3,4,5

Sample Input 2	*Sample Output 2*
6 5 0 1 1 2 2 3 3 4 4 5	Stable Water Supply

Question 4:

A city has a network of **hospitals and intersections**, represented as a **directed graph**. Each **vertex** in the graph represents a **location**, while each **edge** represents a **road** between two locations in a given direction. The **edge weight** represents the **time in minutes** required to travel between two locations.

An **ambulance** stationed at a hospital (starting location) must reach an emergency location (destination) in the shortest possible time.

Your task is to write a program that uses **Dijkstra's algorithm** to determine the **fastest route** from the hospital to the emergency location.

The input: are stored in the ***input.txt*** text file:

- The **first line** contains **two integers**:
  - N → The number of locations (vertices).
  - M → The number of roads (edges).
- The next **M** lines contain **three integers**:
  - **A B T**, where **A** and **B** are locations, and **T** is the **time in minutes** to travel from **A** to **B**.
- The last line contains **two integers**:
  - **Start and End**, representing the **hospital location (starting point)** and the **emergency location (destination)**.

The output: the results need to be saved to the ***output.txt*** text file:

- If there is a path from the warehouse to the customer's location, output 2 lines:
  - 1st line: The shortest time (in minutes) from the hospital to the emergency location.
  - 2nd line: The fastest route from the hospital to the emergency location.
- If there is no route, output "No path".

Sample Input 1	*Sample Output 1*
6 8 0 1 4 0 2 1 1 3 1 2 3 5 1 4 3 4 5 2 3 5 2 2 4 7 0 5	7 0->1->3->5

Sample Input 2	*Sample Output 2*
5 6 0 1 3 1 2 5 0 2 2	7 0->2->4

2 3 6 3 4 2 2 4 5 0 4	

Sample Input 3	*Sample Output 3*
6 4 0 1 2 1 2 3 3 4 1 4 5 5 0 5	No path

Question 5:

A city has **N buildings** numbered from **0 to N-1**. Some buildings are connected by **fiber-optic cables**, forming an **undirected graph**. Each cable has a **latency (in milliseconds)** representing the time taken for data to travel between two buildings.

The **main Internet hub** of the city is located in **building 0**, and the city administration wants to ensure that the **fastest connection** to a **target building** is determined.

Your task is to write a program to find the **fastest Internet connection** from the central hub (building **0**) to the **target building**..

The input: are stored in the ***input.txt*** text file:

- The **first line** contains **two integers**:
  - N → The number of buildings (vertices).
  - M → The number of fiber-optic cables (edges).
- The next **M** lines contain **three integers**:
  - **A B T**, where **A** and **B** are locations, and **T** is the **time in minutes** to travel from **A** to **B**.
  - **A B L**, where:
    - **A** and **B** are two buildings.
    - **L** is the **latency (in milliseconds)** of the fiber-optic cable between **A and B** (since the graph is **undirected**, **A is connected to B and vice versa**).
- The last line contains **two integers**:
  - Start and End, representing the Internet hub (starting point) and target building (destination).

The output: the results need to be saved to the ***output.txt*** text file:

- If there is a path from the warehouse to the customer's location, output 2 lines:
  - 1st line: The **minimum latency** (in milliseconds) from the Internet hub to the target building..
  - 2nd line: The **shortest route** from the hub to the target building.
- If **no connection exists**, output "No connection".

Sample Input 1	*Sample Output 1*
6 7 0 1 10 0 2 5 1 3 6 2 3 2 1 4 8 3 5 4 4 5 3 0 5	11 0->2->3->5

Sample Input 2	*Sample Output 2*
6 8	6

0 1 3 0 2 1 1 3 5 2 3 2 1 4 4 4 5 2 3 5 3 2 4 7 0 5	0->2->3->5

Sample Input 3	*Sample Output 3*
6 4 0 1 2 1 2 3 3 4 1 4 5 5 0 5	No connection

Question 6:

A tourist is visiting a city with **N locations** (numbered **0 to N-1**) and a set of **one-way roads** connecting some locations. The tourist starts at a **known location** but has **gotten lost** while trying to reach a specific destination.

Your task is to **help the tourist find the shortest path** from their current location to their destination. If there is no way to reach the destination, notify the tourist.

The input: are stored in the ***input.txt*** text file:

- **The first line** contains two integers:
  - **N** → Number of locations (**vertices**).
  - **M** → Number of one-way roads (**edges**).
- **The next M lines**, each containing three integers:
  - **A B D,** where:
    - **A** and **B** are two locations.
    - **D** is the **distance** (in kilometers) of the one-way road from **A to B**.
- **The last line** contains two integers:
  - **Start and End**, **representing** the **tourist's current location** and the **destination**.

The output: the results need to be saved to the ***output.txt*** text file:

- If there is a path from the warehouse to the customer's location, output 2 lines:
  - 1st line: The **minimum distance** (in kilometers) to reach the destination.
  - 2nd line: The **shortest path** taken.
- If **no path exists**, output "Lost".

Sample Input 1	*Sample Output 1*
6 7 0 1 5 0 2 3 1 3 4 2 3 2 1 4 6 3 5 3 4 5 2 0 5	8 0->2->3->5

Sample Input 2	*Sample Output 2*
4 2 0 1 1 1 2 2 3 2	Lost

Sample Input 3	*Sample Output 3*
7 9 0 1 2	11 3->4->5->6

0 2 4 1 3 3 2 3 1 3 4 6 4 5 2 5 6 5 1 6 12 2 6 10 3 6	

Question 7:

A country has **N islands** (numbered from **0 to N-1**). The islands are connected by **bridges**, allowing transportation between them. However, after a **severe storm**, some bridges were destroyed, isolating certain islands and making travel between them impossible.

The government wants to **rebuild the minimum number of bridges** to ensure that **all islands are connected** again.

Your task is to write a program that calculates the **minimum number of bridges that need to be rebuilt**.

The input: are stored in the ***input.txt*** text file:

- **The first line** contains two integers:
  - ***N:*** Number of islands (**vertices**).
  - ***M:*** Number of existing bridges (**edges**).
- **The next M lines**, each containing two integers **u v**, represent a bridge connecting island **u** and island **v**.

The output: the results need to be saved to the ***output.txt*** text file:

- A single integer representing the **minimum number of bridges** that need to be rebuilt to ensure all islands are connected.

Sample Input 1	*Sample Output 1*
7 5 0 1 1 2 2 3 4 5 5 6	1

Sample Input 2	*Sample Output 2*
9 5 0 1 0 2 3 4 5 6 7 8	3

Sample Input 3	*Sample Output 3*
6 6 0 1 1 2 2 3 3 4 4 5 0 5	0

Question 8:

A city has **N districts** numbered from **0 to N-1**. The city administration wants to deploy a **fiber optic network** to connect all districts in such a way that the **total installation cost is minimized**.

You are given a list of possible fiber optic connections between districts, along with the **installation cost** of each connection. Your task is to determine the **set of fiber optic connections** that need to be established so that all districts are connected with the **minimum total cost**.

Your task is to write a program to determine the most cost-effective way to connect all locations.

The input: are stored in the ***input.txt*** text file:

- **The first line** contains two integers:
  - **N** ($1 \leq N \leq 1000$) → Number of districts.
  - **M** ($1 \leq M \leq 10^5$) → Number of possible fiber optic connections.
- **The next M lines**, each containing three integers **u v w**:
  - **u, v** ($0 \leq u, v < N$) → Two districts that can be connected.
  - **w** ($1 \leq w \leq 10^6$) → Installation cost between **u and v**.

The output: the results need to be saved to the ***output.txt*** text file:

- A single integer: **The minimum total cost to connect all districts**.
- If it is not possible to connect all districts, print **"Impossible"**.

Sample Input 1	*Sample Output 1*
5 7 0 1 2 0 2 3 1 2 1 1 3 4 2 3 5 2 4 6 3 4 7	13

Sample Input 2	*Sample Output 2*
6 4 0 1 5 1 2 10 3 4 2 4 5 3	Impossible

Question 9:

A country has **N cities** numbered from **0 to N-1**. The government is planning to **construct a power grid** to **connect all cities** with **the lowest cost possible**.

Some power line routes have already been surveyed, and each route has a **different construction cost**. Your task is to determine **the set of power lines to be built** so that all cities are connected with **the minimum total cost**.

Your task is to write a program to determine a subset of connections that ensures all locations are linked while minimizing the total cost.

The input: are stored in the ***input.txt*** text file:

- **The first line** contains two integers:
  - **N** ($1 \leq N \leq 1000$) → Number of cities.
  - **M** ($1 \leq M \leq 10^5$) → Number of possible power lines.
- **The next M lines**, each containing three integers **u v w**:
  - **u, v** ($0 \leq u, v < N$) → Two cities that can be connected.
  - **w** ($1 \leq w \leq 10^6$) → Construction cost of the power line between **u and v**.

The output: the results need to be saved to the ***output.txt*** text file:

- A single integer: **The minimum total cost to connect all cities**.
- If it is not possible to connect all cities, output **"Impossible"**.

Sample Input 1	*Sample Output 1*
6 9 0 1 4 0 2 4 1 2 2 1 3 5 2 3 1 2 4 3 3 4 8 3 5 6 4 5 7	16

Sample Input 2	*Sample Output 2*
5 3 0 1 10 2 3 5 3 4 8	Impossible