Programming & Project Design

To: The New Logistics Team From: Management, Global Disaster Solutions Inc.

Subject: Urgent Deployment

Congratulations! Your team has just been promoted to Head of Logistics at Global Disaster Solutions. Why? largely because the previous team looked at the weather forecast and immediately quit.

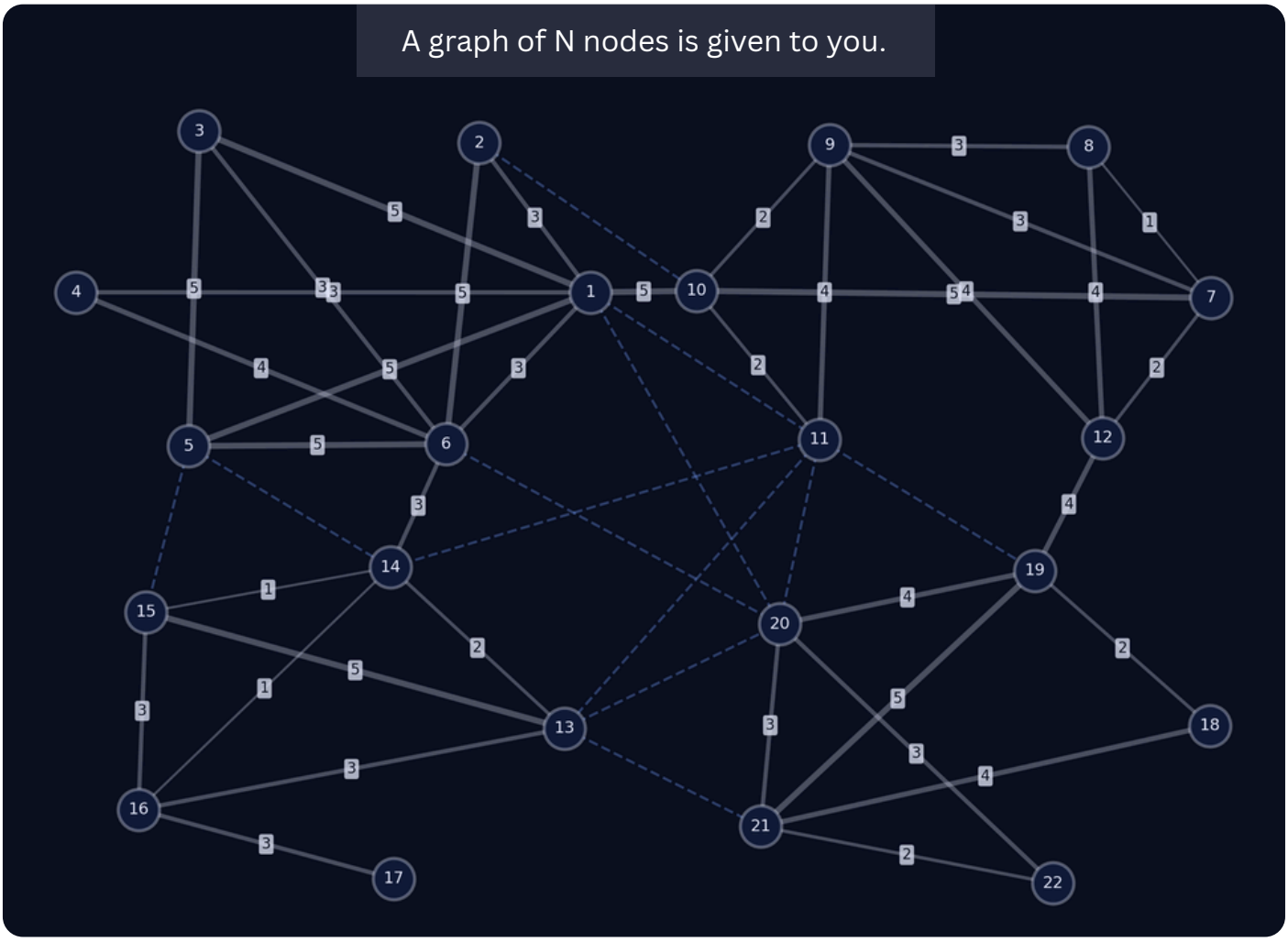
The city of **Polyville** is having a really bad day. We are talking about a "Biblical Plague" level event. The ground is shaking, the sky is leaking, and the wind is blowing hard enough to peel paint. To make matters worse, our sensor network is reporting that the road conditions are changing every single minute.

You have a fleet of **Trucks** (which are terrified of mudslides) and **Drones** (which are easily blown away). Your job is simple: Write an algorithm to guide these vehicles through the chaos, deliver the supplies, and try not to destroy millions of dollars worth of company property.

Regards,
CEO

Coding Club

2 The Environment



Time: The simulation proceeds in discrete ticks (t=0, 1, 2, ...,Tmax).
Adjacency Matrix: You are given an N X N matrix defining the connections between nodes.

- -1: No Connection.
- 0: Airspace (Passable by Drones Only).
- 1, 2, 3, 4, 5: Surface Roads (Passable by Trucks & Drones).

Your Fleet:

- **Trucks:** Can only traverse edges with Road Types **1 through 5.**
- **Drones:** Can traverse edges with Road Types **0 through 5.**
- **Start:** All vehicles begin at Node 1.
- **Movement:**
  - Moving between connected nodes takes **1 Time Tick**.
  - **Waiting:** A vehicle can stay at its current node. This takes 1 tick and incurs **0 Cost.**
  - **Multiple** trucks and drones **may share the same space.**

3 Mechanics: Costs & Blocking

The cost to traverse a road is dynamic. It depends on the **Road Type**, the **Base Weight at that time**, and **the Weather Conditions.**

3.1 Base Weight (W_base)

Every Road Type (1-5) has a specific "Base Weight" that changes over time. You are provided a lookup table in the map file: road_weights[Type][Time]. (Note: Road Type 0 (Airspace) has a constant Base Weight of 0).

3.2 Weather Sensors & Blocking (The "Perfect Storm" Rule)

At every time step, 4 sensor readings are provided: Wind, Visibility, Earth Shock, Rainfall. A road is considered "Blocked" (Hazardous) only if BOTH relevant weather factors exceed their safety thresholds. **(You can go through blocked road at a higher cost)**

1. For Trucks (Roads 1-5)

Trucks are susceptible to ground instability and mudslides. A road is blocked for a Truck if **both** Earth Shock and Rainfall are **too high**:

$$(W_{base} \times S_{earth} > 10) \quad \textbf{AND} \quad (W_{base} \times S_{rain} > 30)$$

2. For Drones (Roads 0-5)

Drones are susceptible to air turbulence and blindness. A road is blocked for a Drone if Wind is **too high** AND Visibility is **too low**:

$$(W_{base} \times S_{wind} > 60) \quad \textbf{AND} \quad (W_{base} \times S_{vis} < 6)$$

3.3 Calculating Traversal Cost

When a vehicle moves from Node A to Node B at time t:

1. **Check Status:** Determine if the edge is Safe or Blocked using the "AND" rule above.
2. **Apply Cost Formula:**
   - If Safe:
     - Cost = W_{base}(Type, t) X {RoadType}
   - If Blocked (Hazardous):
     - Cost = W_{base}(Type, t) X {RoadType} **X 5**
   -

Note on Road Type 0 (Airspace): These aerial roads are only accessible to drones. These roads are never blocked irrespective of the sensor data. You are permitted to cross these roads without incurring any cost.

4 Objectives & Scoring

You receive a list of Objectives.

Each objective has:

- **Target Node:** Node target
- **Start Time:** Tstart
- **Deadline:** Tend
- **Reward:** Pmax

A vehicle must arrive at Node target at time tarrival .

- **Valid Window:** You can only complete the objective if **Tstart ≤ tarrival ≤ Tend.**
- You can wait at a node at no additional cost.

Objective Score Calculation

- **Perfect Timing** (tarrival = Tstart): Score = Pmax
- **Late Arrival** (tarrival > Tstart): Points decay linearly. (increment specified in objectives.json.
- **Score**=Pmax - (late_penalty_per_step) * (tarrival - tstart)
- **Missed** (tarrival > Tend): Score = 0

Final Score Formula

Your goal is to maximize the Total Score:

$$\text{Total Score} = \sum(\text{Objective Scores}) - \sum(\text{Travel Costs Incurred})$$

5 Files & Output

Input Files

1. public_map.json
- N: Number of nodes.
- T: Max time steps.
- map: The Adjacency Matrix (Road Types).
- road_weights: Dictionary mapping Road Types ("1", "2"...) to an array of weights for each time step.

2. sensor_data.json
- Contains arrays of length T for environmental factors:
  - rainfall
  - wind
  - visibility
  - earth_shock
-

3. objectives.json
- start_node: Where everyone begins.
- late_penalty_per_step: The cost of being late.
- objectives: A list of objects containing id, node, release, deadline, and points.

Output Format

Your program must be able to create a single JSON file mapping each vehicle to its path.
- Key: Vehicle ID (truck1, truck2, ... and drone1, drone2, ...).
- Value: List of Node IDs visited at each time step.

Example JSON

```
{
 "truck1": [1, 5, 6, 6, 12],
 "truck2": [1, 1, 2, 8, 4],
 "drone1": [1, 10, 15, 20, 25]
}
```

- Path length must equal T = 100.
- path[t] to path[t+1] must be a valid connection.
- Trucks cannot fly (No Road Type 0).

6 Submission

A **zipped folder** of the below file structure:

- folder/
  - main.py
  - requirements.txt
  - all_input_files*.json
  - modules/
    - my_custom_module_1.py
    - custom2.py

Your solution will be executed with:

Input Rules

- All input files will be placed in the root folder.
- Read data only from the provided files.

Output Rules

- Your program must generate a file named solution.json.
- The file must be saved in the root folder (same location as main.py).

Development Constraints

- Any extra logic, helpers, or scripts must be inside the modules/ folder.
- main.py should act as the entry point and coordinate the workflow.
- Avoid hardcoded paths or assumptions.

What We Evaluate

We are not checking only correctness.
We value:

- Ability to handle large-scale inputs
- Efficient time and memory usage
- Clean, modular architecture
- Readable and maintainable code

"The rain doesn't fall. It renders."- Shaman

"Do not pray to the storm. Pray to the Admin." -Shaman

Page not found

"Tryst Trial Mist, List Fist Dial,Wrist Twist File, Gist Hissed Vile,Missed Kissed Compile, Cyst Twixt Tile,Script Stripped Smile, Resist Desist Denial" - Shaman