Delhi Metro Route and Schedule Simulator

Objective

You are required to design and implement a **Python program** that simulates the working of the **Delhi Metro system**, focusing on the **Blue Line** and **Magenta Line**.

Your program should:

1. Load and store the metro stop data (station names, order, and travel times).
2. Compute when the **next metro** will arrive at any given station.
3. Allow a **rider to plan a journey**, including **interchange** between Blue and Magenta lines.

Phase 1: Data Collection

Before writing any code, you must **collect and structure real-world data** for the Delhi Metro.

Lines to Include

- **Blue Line** (Dwarka Sector 21 ↔ Noida Electronic City / Vaishali)
- **Magenta Line** (Janakpuri West ↔ Botanical Garden)

Data to Collect for Each Line

Attribute	Description
Station Order	All station names in correct travel order
Travel Time	Approximate time (in minutes) between consecutive stations
Interchange Points	Stations where transfer between Blue and Magenta is possible (e.g., *Janakpuri West* and *Botanical Garden*)

You can collect data from:

- The **official DMRC website**

Deliverable

Create a data file in **text format**, **i.e. as metro_data.txt**

Include all related information which you think is required for your program. For example:

Line,	Station,	NextStation,	TravelTime(min),	Interchange Point
Blue,	Dwarka Sector 21,	Dwarka,	3	No
Blue,	Dwarka,	Rajouri Garden,	5	No
Magenta,	Janakpuri West,	Palam,	4	Yes

You may save it as you feel is easy/efficient for you. There is no fixed

---

Phase 2: Program Functionality

In this phase you will implement two functionalities:

1. **Metro Timings Module**: In this part, you will load your station and travel-time data to calculate when the next train will arrive at any station based on the current time.
2. **Ride journey planner**: Here, you will build a program that finds the best route between two stations, showing total travel time, interchange points, and when the rider can catch the next train.

Metro Timings

- **Start time:** 06:00 AM
- **End time:** 11:00 PM
- **Frequency:**
  - Off-peak hours: every **8 minutes**
  - Peak hours (8–10 AM and 5–7 PM): every **4 minutes**

The program must compute when the **next metro** will arrive at a given station based on current time. This information will be inferred from the file you have created in the first phase.

Note: you will not use any libraries such as pickle, pandas, numpy etc. You are allowed Only basic file modes (read, write, append).

Example:

```
User Input:
Line = Blue
Station = Rajiv Chowk
Current time = 09:18

Output:
Next metro at 09:20
Subsequent metros at 09:24, 09:28, ...
```

Rider Journey Planner

```
User Input:
Source: Dwarka
Destination: Botanical Garden
Time of travel: 08:22
```

Your program should:

1. Identify whether the route is on a single line or requires interchange.
2. Calculate:
    - Next available metro at source
    - Estimated arrival time at each station
    - If an interchange is needed, determine:
        - Where to change (e.g., *Janakpuri West*)
        - Next metro on the connecting line based on arrival time
    - Final arrival time at destination

Example Output:

```
Journey Plan:
Start at Dwarka (Blue Line)
Next metro at 08:24
Arrive at Janakpuri West at 08:39
Transfer to Magenta Line
Next Magenta metro departs at 08:42
Arrive at Botanical Garden at 09:15
Total travel time: 51 minutes
```

Service Restrictions

Your program may handle:

- Display “No service available” if time is before **06:00 AM** or after **11:00 PM**
- Add **a few minutes interchange delay** at transfer stations depending on the time provided on the official website.

Optional Enhancements: Bonus

- Add more lines (e.g., Yellow or Green Line)
- Include **fare calculation** based on travel distance or time
- Any other functionality, which you feel is relevant.

Phase 3: Documentation & Submission

Each student must submit:

1. **Python Source Code:** `metro_simulator.py`
2. **Data File:** `metro_data.txt`
3. **README File:** Include

    - Data sources
    - Assumptions made
    - Instructions to run the program
4. ***(Optional)*** A screenshot or short demo video

Evaluation Criteria

Component	Marks	Description
Data Collection	20	Accuracy and completeness of Blue & Magenta line data
Schedule Logic	30	Correct computation of next train times
Journey Planner	25	Accurate routing, interchange handling, timing logic
Code Quality	15	Modularity, readability, error handling
Documentation	10	Clear explanations and usage guide
Extensions	20	More lines, fare calculations
