Weather Data Storage System – Assignment Report

1. Problem Understanding

The task is to develop a Weather Data Storage System that can store, retrieve, and manage weather data for different cities and years. The program should be able to insert, delete, and retrieve weather records. It must use both a dense 2D array storage system and a sparse data representation. Additionally, row-major and column-major access methods are implemented to compare efficiency. Finally, time and space complexity of key operations must be analyzed.

2. Weather Record ADT

Attributes:

1. Date (LocalDate)
2. City (String)
3. Temperature (double)

Methods:

1. insert(data): Insert a record
2. delete(data): Delete a record
3. retrieve(city, year): Retrieve record

3. Data Storage System

Dense Storage (2D Array):

1. Rows = years
2. Columns = cities
3. Stores temperatures as double[][]
4. Missing values handled with sentinel Double.NaN

Sparse Storage (HashMap):

1. Key = (year-city)
2. Value = temperature
3. Only non-empty values stored → memory efficient

4. Row-Major vs Column-Major Access

Row-Major: Iterates row by row (year first). Column-Major: Iterates column by column (city first). On Java arrays (array of arrays), row-major is generally faster due to cache locality.

5. Time and Space Complexity

1. Insert: O(1)
2. Delete: O(1)
3. Retrieve: O(1)
4. Row/Column traversal: O(R × C)
5. Space (Dense): O(R × C)
6. Space (Sparse): O(K), where K is number of records

6. Assignment Coverage

✔ WeatherRecord ADT ✔ Dense 2D Array storage ✔ Row-Major & Column-Major traversal ✔ Sparse data handling ✔ Insert, Delete, Retrieve functions ✔ Complexity analysis ✔ Interactive input for user testing

7. Conclusion

The Weather Data Storage System in Java fulfills all the requirements of the assignment. It demonstrates use of ADTs, 2D arrays, sparse storage, traversal efficiency, and complexity analysis. The system can be extended to include more cities, multiple daily records, and file-based storage.