Develop the weather data storage system
Theme: Trees and Graph Algorithms

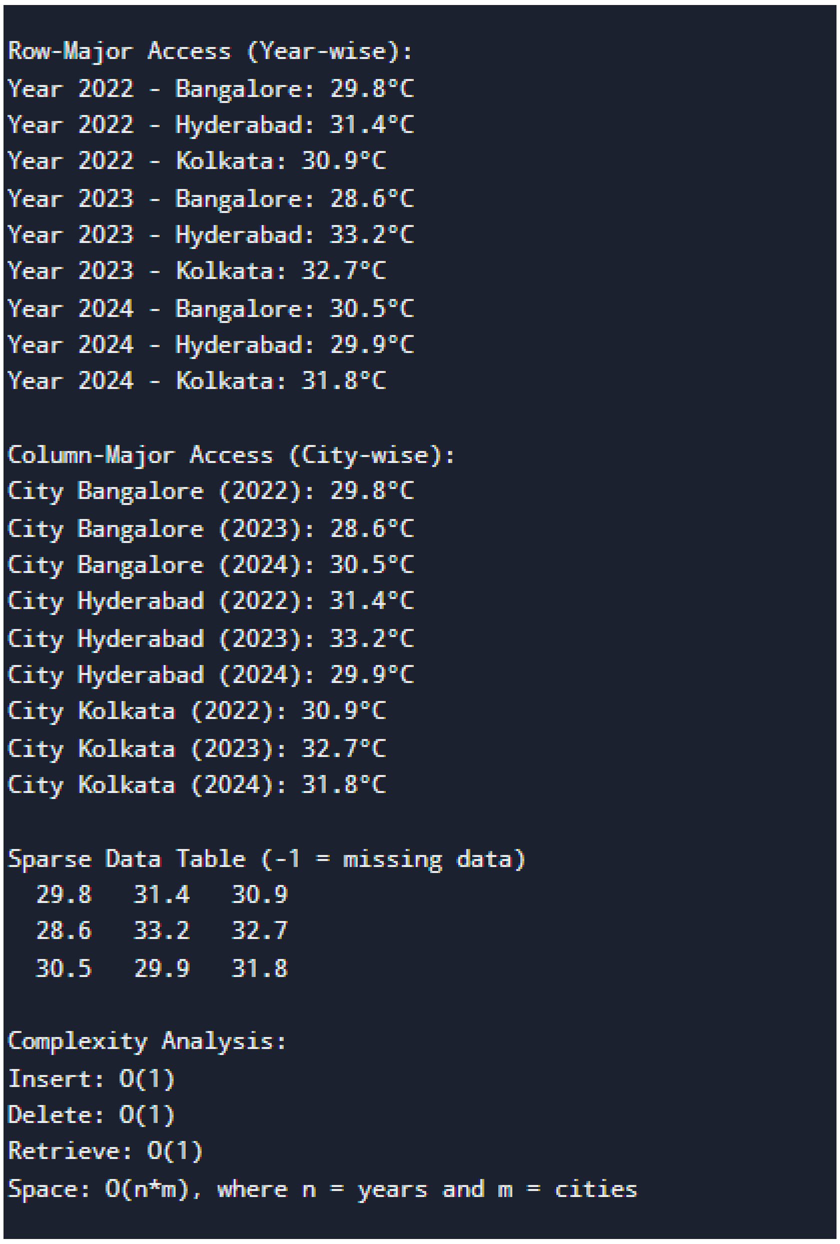
GitHub:

https://github.com/mayank24000/University_Assignments/tree/main/Ds_%20Assignments/Theory_Assignments/Theory_Assignment_1

Code:

```
#include <iostream>
#include <iomanip>
#include <string>
using namespace std;

// Structure to store one record of weather data
struct WeatherRecord {
    string date;
    string city;
    double temperature;
};

// Class to manage the weather storage systems
class WeatherSystem {
private:
    static const int TOTAL_YEARS = 3;
    static const int TOTAL_CITIES = 3;
    double tempData[TOTAL_YEARS][TOTAL_CITIES];
    string cities[TOTAL_CITIES];
    int yearList[TOTAL_YEARS];

public:
    WeatherSystem() {
        cities[0] = "Bangalore";
        cities[1] = "Hyderabad";
        cities[2] = "Kolkata";

        yearList[0] = 2022;
        yearList[1] = 2023;
        yearList[2] = 2024;

        // initialize all temperature values to -1 (means no record)
        for (int i = 0; i < TOTAL_YEARS; i++) {
            for (int j = 0; j < TOTAL_CITIES; j++) {
                tempData[i][j] = -1;
            }
        }
    }

    // Insert temperature data for all years and cities
    void fillData() {
        tempData[0][0] = 29.8;
        tempData[0][1] = 31.4;
        tempData[0][2] = 30.9;

        tempData[1][0] = 28.6;
        tempData[1][1] = 33.2;
        tempData[1][2] = 32.7;

        tempData[2][0] = 30.5;
        tempData[2][1] = 29.9;
        tempData[2][2] = 31.8;
    }

    // Row-major access (year by year)
    void accessByRow() {
        cout << "\nRow-Major Access (Year-wise):\n";
        for (int i = 0; i < TOTAL_YEARS; i++) {
            for (int j = 0; j < TOTAL_CITIES; j++) {
                cout << "Year " << yearList[i] << " - " << cities[j]
                     << ": " << tempData[i][j] << "°C" << endl;
            }
        }
    }

    // Column-major access (city by city)
    void accessByColumn() {
        cout << "\nColumn-Major Access (City-wise):\n";
        for (int j = 0; j < TOTAL_CITIES; j++) {
            for (int i = 0; i < TOTAL_YEARS; i++) {
                cout << "City " << cities[j] << " (" << yearList[i]
<< "): "
                     << tempData[i][j] << "°C" << endl;
            }
        }
    }

    // Show sparse data table
    void showSparseData() {
        cout << "\nSparse Data Table (-1 = missing data)\n";
        for (int i = 0; i < TOTAL_YEARS; i++) {
            for (int j = 0; j < TOTAL_CITIES; j++) {
                cout << setw(6) << tempData[i][j] << " ";
            }
            cout << endl;
        }
    }

    // Display basic time and space complexity
    void showComplexity() {
        cout << "\nComplexity Analysis:\n";
        cout << "Insert: O(1)\n";
        cout << "Delete: O(1)\n";
        cout << "Retrieve: O(1)\n";
        cout << "Space: O(n*m), where n = years and m = cities\n";
    }
};

int main() {
    WeatherSystem ws;

    ws.fillData();
    ws.accessByRow();
    ws.accessByColumn();
    ws.showSparseData();
    ws.showComplexity();

    return 0;
}
```

OUTPUT:


```
Row-Major Access (Year-wise):
Year 2022 - Bangalore: 29.8°C
Year 2022 - Hyderabad: 31.4°C
Year 2022 - Kolkata: 30.9°C
Year 2023 - Bangalore: 28.6°C
Year 2023 - Hyderabad: 33.2°C
Year 2023 - Kolkata: 32.7°C
Year 2024 - Bangalore: 30.5°C
Year 2024 - Hyderabad: 29.9°C
Year 2024 - Kolkata: 31.8°C

Column-Major Access (City-wise):
City Bangalore (2022): 29.8°C
City Bangalore (2023): 28.6°C
City Bangalore (2024): 30.5°C
City Hyderabad (2022): 31.4°C
City Hyderabad (2023): 33.2°C
City Hyderabad (2024): 29.9°C
City Kolkata (2022): 30.9°C
City Kolkata (2023): 32.7°C
City Kolkata (2024): 31.8°C

Sparse Data Table (-1 = missing data)
  29.8   31.4   30.9
  28.6   33.2   32.7
  30.5   29.9   31.8

Complexity Analysis:
Insert: O(1)
Delete: O(1)
Retrieve: O(1)
Space: O(n*m), where n = years and m = cities
```