```
#include <iostream>
#include <string>
#include <vector>
#include <iomanip>
#include <algorithm>

using namespace std;

class weatherrecord {
private:
    string date;
    string city;
    double temperature;

public:
    weatherrecord(string d, string c, double temp) {
        date = d;
        city = c;
        temperature = temp;
    }

    string getdate() { return date;}
    string getcity() { return city;}
    double gettemperature() { return temperature;}

    void displayrecord(){
        cout << "Date : " << date << " | City : " << city << " |
Temperature : " << temperature << "Â°C" << endl;
    }
};

class weatherstorage {
private:
    vector <vector<double>> temperaturedata ;
    vector <string> years;
    vector <string> cities;
    const double SENTINEL_VALUE = -999.0;
public:
    weatherstorage(vector<string> yearlist, vector<string> citylist) {
        years = yearlist;
        cities = citylist;
        temperaturedata.resize(years.size(),
vector<double>(cities.size(), SENTINEL_VALUE));
    }
    void insertdata(string date, string city, double temp) {
        int yearindex = findyearindex(date);
        int cityindex = findcityindex(city);
        if ( yearindex != -1 && cityindex != -1) {
            temperaturedata[yearindex][cityindex] = temp;
            cout << " Data inserted successfully ! " << endl;
        } else {
            cout << " Error : year / city not found in storage " << endl;
        }
    }
    void insertdata(string date, string city) {
        int yearindex = findyearindex(date);
        int cityindex = findcityindex(city);
        if (yearindex != -1 && cityindex != -1) {
```

```
            temperaturedata[yearindex][cityindex] = SENTINEL_VALUE;
            cout << " Data deleted successfully ! " << endl;
        } else {
            cout << " Error : data not found " << endl;
        }
    }
    double retrivedata(string year, string city) {
        int yearindex = findyearindexfromyear(year);
        int cityindex = findcityindex(city);
        if (yearindex != -1 && cityindex != -1) {
            return temperaturedata[yearindex][cityindex];
        }
        return SENTINEL_VALUE;
    }
    void populatearray() {
        insertdata("15/01/2020", "Delhi", 25.5);
        insertdata("15/01/2021", "Delhi", 26.0);
        insertdata("15/01/2020", "Mumbai", 30.2);
        insertdata("15/01/2021", "Mumbai", 31.0);
        insertdata("15/01/2020", "Chennai", 28.7);
        cout << "Array populated with sample data!" << endl;
    }
    void rowmajoraccess() {
        cout << "\n === ROW MAJOR ACCESS === " << endl;
        for ( int i = 0; i < years.size(); i++) {
            cout << "Year " << years[i] << " : ";
            for (int j = 0; j < cities.size(); j++) {
                if ( temperaturedata[i][j] != SENTINEL_VALUE) {
                    cout << cities[j] << "( " << temperaturedata[i][j] <<
" )";
                } else {
                    cout << cities[j] << "(NO DATA)" ;
                }
            }
            cout << endl;
        }
    }
    void columnmajoraccess() {
        cout << "\n === COLUMN MAJOR ACCESS === " << endl;
        for (int j = 0; j < cities.size(); j++) {
            cout << "Year " << cities[j] << " : ";
            for (int i = 0; i < years.size(); i++) {
                if (temperaturedata[i][j] != SENTINEL_VALUE) {
                    cout << years[j] << "( " << temperaturedata[i][j] <<
")" ;
                } else {
                    cout << years[j] << "(NO DATA)";
                }
            }
            cout << endl;
        }
    }
    void handlesparsedata() {
        cout << "\n=== SPARSE DATA REPRESENTATION ===" << endl;
        cout << "Non-empty cells (row, column, temperature):" << endl;

        int datacount = 0;
        for ( int i = 0; i < years.size(); i++) {
```

```
            for (int j = 0; j < cities.size(); j++) {
                if (temperaturedata[i][j] != SENTINEL_VALUE) {
                    cout << "[" << i << "," << j << "] = " <<
temperaturedata[i][j] << "Â°C";
                    cout << " (Year: " << years[i] << ", City: " <<
cities[j] << ")" << endl;
                    datacount++;
                }
            }
        }
        if (datacount == 0) {
            cout << "No data unavailable!" << endl;
        } else {
            cout << "Total data points: " << datacount << endl;
        }
    }
    void analyzecomplexity() {
        cout << "\n=== TIME AND SPACE COMPLEXITY ANALYSIS ===" << endl;
        cout << "Insert Operation: O(1) - Constant time (direct array
access)" << endl;
        cout << "Delete Operation: O(1) - Constant time (direct array
access)" << endl;
        cout << "Retrieve Operation: O(1) - Constant time (direct array
access)" << endl;
        cout << "Row/Column Major Access: O(n*m) - Linear time (n years
Ã— m cities)" << endl;
        cout << "Space Complexity: O(n*m) - For n years and m cities" <<
endl;
        cout << "Sparse Data Handling: Saves space by tracking only non-
empty cells" << endl;
    }
    void displayalldata() {
        cout << "\n=== COMPLETE WEATHER DATA ===" << endl;
        cout << setw(10) << "Year" << setw(15) << "City" << setw(15) <<
"Temperature" << endl;
        cout << "----------------------------------------" << endl;
        for ( int i = 0; i < years.size(); i++) {
            for ( int j = 0; j < cities.size(); j++) {
                cout << setw(10) << years[i] << setw(15) << cities[j] <<
setw(15);
                if (temperaturedata[i][j] != SENTINEL_VALUE) {
                    cout << temperaturedata[i][j] << "Â°C";
                } else {
                cout << "No data";
                }
                cout << endl;
            }
        }
    }
private :
    int findyearindex(string date) {
        string year = date.substr(6,4 );
        return findyearindexfromyear(year);
    }
    int findyearindexfromyear(string year) {
        for (int i = 0; i < years.size(); i++) {
            if ( years[i] == year) {
                return i;
```

```
            }
        }
        return -1;
    }
    int findcityindex(string city) {
        for (int i = 0; i < cities.size(); i++) {
            if (cities[i] == city) {
                return i;
            }
        }
        return -1;
    }
};

int main() {
    cout << "=== WEATHER DATA STORAGE SYSTEM ===" << endl;
    vector<string> years = {"2020", "2021", "2022", "2023"};
    vector<string> cities = {"Delhi", "Mumbai", "Chennai", "Kolkata",
"Bangalore"};

    weatherstorage weathersystem(years, cities);

    int choice;
    string date, city, year;
    double temperature;

    do {
        cout << "\n===== WEATHER SYSTEM MENU =====" << endl;
        cout << "1. Populate with Sample Data" << endl;
        cout << "2. Insert Weather Record" << endl;
        cout << "3. Delete Weather Record" << endl;
        cout << "4. Retrieve Temperature" << endl;
        cout << "5. Row-Major Access" << endl;
        cout << "6. Column-Major Access" << endl;
        cout << "7. Handle Sparse Data" << endl;
        cout << "8. Display All Data" << endl;
        cout << "9. Analyze Complexity" << endl;
        cout << "10. Exit" << endl;
        cout << "Enter your choice: ";

        cin >> choice;

        switch(choice) {
            case 1:
                weathersystem.populatearray();
                break;

            case 2:
                cout << "Enter date (DD/MM/YYYY): ";
                cin >> date;
                cout << "Enter city: ";
                cin >> city;
                cout << "Enter temperature: ";
                cin >> temperature;
                weathersystem.insertdata(date, city, temperature);
                break;

            case 3:
```

```
            cout << "Enter date (DD/MM/YYYY): ";
            cin >> date;
            cout << "Enter city: ";
            cin >> city;
            weathersystem.insertdata(date, city);
            break;

        case 4:
            cout << "Enter city: ";
            cin >> city;
            cout << "Enter year: ";
            cin >> year;
            {
                double temp = weathersystem.retrivedata(year, city);
                if (temp != -999.0)
                {
                    cout << "Temperature: " << temp << "Â°C" << endl;
                }
                else
                {
                    cout << "No data found!" << endl;
                }
            }
            break;

        case 5:
            weathersystem.rowmajoraccess();
            break;

        case 6:
            weathersystem.columnmajoraccess();
            break;

        case 7:
            weathersystem.handlesparsedata();
            break;

        case 8:
            weathersystem.displayalldata();
            break;

        case 9:
            weathersystem.analyzecomplexity();
            break;

        case 10:
            cout << "Exiting system. Goodbye!" << endl;
            break;

        default:
            cout << "Invalid choice! Please try again." << endl;
        }
    } while (choice != 10);

    return 0;
}
```