#include <iostream>

#include <vector>

#include <map>

#include <string>

#include <fstream>

#include <sstream>

#include <iomanip>

// Location class with name, latitude, and longitude attributes

Class Location {

Public:

Std::string name;

Double latitude;

Double longitude;

Location(std::string n, double lat, double lon) : name(n), latitude(lat), longitude(lon) {}

};

// WeatherVariable class to manage different weather parameters

Class WeatherVariable {

Public:

Std::string name;

Double value;

WeatherVariable(std::string n, double v) : name(n), value(v) {}

};

// WeatherForecastSystem class to handle API interface and data retrieval (stubbed)

Class WeatherForecastSystem {

Private:

Std::map<std::string, std::vector<WeatherVariable>> forecasts;

Public:

Void fetchForecast(Location location) {

// Simulate fetching weather data (this would be where an API call is made)

Std::vector<WeatherVariable> weatherData;

weatherData.push\_back(WeatherVariable(“Temperature”, 23.5));

weatherData.push\_back(WeatherVariable(“Wind Speed”, 5.2));

forecasts[location.name] = weatherData;

std::cout << “Fetched forecast for “ << location.name << std::endl;

}

Void displayForecast(const Location& location) {

Std::cout << “Weather forecast for “ << location.name << “:\n”;

If (forecasts.find(location.name) != forecasts.end()) {

For (const auto& variable : forecasts[location.name]) {

Std::cout << variable.name << “: “ << variable.value << std::endl;

}

} else {

Std::cout << “No forecast data available.\n”;

}

}

Void exportToCSV(const std::string& filename) {

Std::ofstream file(filename);

File << “Location,Variable,Value\n”;

For (const auto& entry : forecasts) {

For (const auto& variable : entry.second) {

File << entry.first << “,” << variable.name << “,” << variable.value << “\n”;

}

}

File.close();

Std::cout << “Data exported to “ << filename << std::endl;

}

Void exportToJSON(const std::string& filename) {

Std::ofstream file(filename);

File << “{\n”;

For (auto it = forecasts.begin(); it != forecasts.end(); ++it) {

File << “ \”” << it->first << “\”: {\n”;

For (size\_t I = 0; I < it->second.size(); ++i) {

File << “ \”” << it->second[i].name << “\”: “ << it->second[i].value;

If (I < it->second.size() – 1) file << “,”;

File << “\n”;

}

File << “ }”;

If (std::next(it) != forecasts.end()) file << “,”;

File << “\n”;

}

File << “}\n”;

File.close();

Std::cout << “Data exported to “ << filename << std::endl;

}

};

// HistoricalWeatherSystem class to handle fetching historical weather data from an API (stubbed)

Class HistoricalWeatherSystem {

Private:

Std::map<std::string, std::vector<WeatherVariable>> historicalData;

Public:

Void fetchHistoricalData(Location location) {

// Simulate fetching historical weather data

Std::vector<WeatherVariable> weatherData;

weatherData.push\_back(WeatherVariable(“Temperature”, 18.2));

weatherData.push\_back(WeatherVariable(“Wind Speed”, 3.1));

historicalData[location.name] = weatherData;

std::cout << “Fetched historical data for “ << location.name << std::endl;

}

Void displayHistoricalData(const Location& location) {

Std::cout << “Historical weather data for “ << location.name << “:\n”;

If (historicalData.find(location.name) != historicalData.end()) {

For (const auto& variable : historicalData[location.name]) {

Std::cout << variable.name << “: “ << variable.value << std::endl;

}

} else {

Std::cout << “No historical data available.\n”;

}

}

Void exportToCSV(const std::string& filename) {

Std::ofstream file(filename);

File << “Location,Variable,Value\n”;

For (const auto& entry : historicalData) {

For (const auto& variable : entry.second) {

File << entry.first << “,” << variable.name << “,” << variable.value << “\n”;

}

}

File.close();

Std::cout << “Data exported to “ << filename << std::endl;

}

Void exportToJSON(const std::string& filename) {

Std::ofstream file(filename);

File << “{\n”;

For (auto it = historicalData.begin(); it != historicalData.end(); ++it) {

File << “ \”” << it->first << “\”: {\n”;

For (size\_t I = 0; I < it->second.size(); ++i) {

File << “ \”” << it->second[i].name << “\”: “ << it->second[i].value;

If (I < it->second.size() – 1) file << “,”;

File << “\n”;

}

File << “ }”;

If (std::next(it) != historicalData.end()) file << “,”;

File << “\n”;

}

File << “}\n”;

File.close();

Std::cout << “Data exported to “ << filename << std::endl;

}

};

// Main program

Int main() {

Location loc1(“City A”, 34.0522, -118.2437);

Location loc2(“City B”, 40.7128, -74.0060);

WeatherForecastSystem forecastSystem;

HistoricalWeatherSystem historicalSystem;

// Fetch and display weather forecast

forecastSystem.fetchForecast(loc1);

forecastSystem.displayForecast(loc1);

forecastSystem.exportToCSV(“forecast.csv”);

forecastSystem.exportToJSON(“forecast.json”);

// Fetch and display historical weather data

historicalSystem.fetchHistoricalData(loc2);

historicalSystem.displayHistoricalData(loc2);

historicalSystem.exportToCSV(“historical.csv”);

historicalSystem.exportToJSON(“historical.json”);

return 0;

}