European Vacation

Your team is to write a program that reads data and stores it in a data structure of your choice. Your program must have the ability to hold the names of at least 20 European cities (the initial data contains 11 cities) and their corresponding traditional food items (up to six per city). Your program will be used to allow prospective European travelers to plan their summer vacation. Your team will need to provide the ability to modify information related to the traditional food items (such as adding new food items, changing a price of a food item, or deleting a food item).

Displaying the initial information:

1. Display the list of European cities and their distances from Rome.

2. Display all the traditional food items for any given city (just how one city).

Planning a trip:

3. Provide the capability to visit the initial 11 European cities starting at Berlin.
    a. Plan the trip starting at Berlin then visit each of the other European cities in the most efficient order (recursively choose the city closest to the previous city)
    b. Display the total distance traveled

4. Offer the option to plan the **shortest** trip starting at Paris
    a. Obtain the “number” of European cities to visit
    b. Visit the “number” of cities specified (including Paris)
    c. Plan the trip starting at London then visit the closest city to Paris, then visit the closest city to that city and so on (recursively choose the city closest to the previous city)
    d. Display the total distance traveled
    e. Allow the traveler to purchase multiple traditional food items when visiting the European cities

5. Offer the option to plan a custom trip
    a. Allow a traveler to select the starting European city they wish to visit
    b. Then allow a traveler to select <u>all</u> other European cities they wish to visit
    c. Plan the trip starting with the selected city then visit each of the other European cities in the most efficient order (recursively choose the city closest to the previous city).
    d. Display the total distance traveled
    e. Allow the traveler to purchase multiple traditional food items when visiting the European cities

6. When taking <u>any</u> trip:
    a. A traveler can purchase multiple traditional food items
    b. Your team must keep track of the number of traditional food items purchased at each city
    c. Display the total amount spent at each European city and a grand total for all cities visited

7. Maintenance (administrator only - requires a password to gain access)
    a. Provide the capability to add new European cities and their corresponding food items by having your program read from the input file given to the class
    b. Provide the capability to change the prices of the traditional food items
    c. Provide the capability to add new traditional food items
    d. Provide the capability to delete traditional food items

8. Provide the capability to visit the 13 European cities starting at Berlin.
    a. Plan the trip starting at Amsterdam then visit each of the other European cities in the most efficient order (recursively choose the city closest to the previous city)
    b. Allow the traveler to purchase multiple traditional food items when visiting the European cities

European Vacation

Please let me know your partners by September 7th (three points will be deducted from your scores if you do not meet this deadline). All projects are due by October 19th. **No late projects will be accepted.** Your team must demonstrate your project to me before it will be graded. Each teammate must identify their accomplishments on the project. Not all team members will necessarily earn the same score.

1. Design a very readable, easy to use interface to demonstrate your program.
2. Contingency handling should include addressing invalid input.
3. Write at least 10 agile stories (including description, tasks, test scenarios, and story points) before any software is developed. The team must follow the Scrum process (the Scrum master **must** document all meetings and the product owner must document the backlog).
4. Submit the following UML class diagrams
    a. Three use cases
    b. One activity diagram
    c. Three state diagrams with your project.
5. Submit a test plan.
6. All changes must be persistent between executions.
7. Submit a discussing the **Big-Oh** of your project for at least **five** methods.
8. Identify all the data structures used
9. Each team must use a version control system (only team members should have access to their repository), graphical user interface tool such as QT, automated documentation tool (DOXYGEN), and an Agile management tool (Twilio, GITHUB, etc.), and database software (e.q. SQLite)

Schedule:
First checkpoint – September 21st – 4 points
Second checkpoint – October 5th – 4 points
Final checkpoint – October 19th - 92 points

The project will be graded using the following scale:

Description	Value
Checkpoint 1	4
Checkpoint 2	4
Meet requirements	70
Use of multiple C++ data structures (1 points per data structure)	3
User interface	2
Test Plan	3
Adherence to Scrum/Team Rules	5
UML	3
DOXYGEN	1
Big O (1 point for each method)	5
Total	100
Continuous Integration (extra credit)	3
Total with extra credit	103

Final demonstration meeting:

1. Be prepared to demonstrate all project's requirements within a 20-minute timeframe.
2. All team members must be present.
3. Demonstrate DOXYGEN and Agile management tool
4. Each teammate must identify their accomplishments on the project and assess their teammates via e-mail.
5. Submit all your project artifacts
   a. Code, test plan, agile stories, scrum log, coding standards, team rules, UML diagrams (class, use cases, state diagrams), data structures used, Big Oh analysis