Your agile team is to write an application that stores the given National Football League (NFL) information into an ordered or unordered map (not using the map or unordered map STL or the QT map). The underlying data structure of the map is the choice of the team. The map should store either the stadiums and/or the souvenirs. The application will allow football fans to plan their dream vacation.

1. Write at least 10 agile stories (including description, tasks, test scenarios, and story points) before any software development.
2. Display all the information related to only one particular football team (team name, stadium name, seating capacity, location, conference, division, surface type, stadium roof type, star player). Do not display information related to the other NFL teams.
3. Display the list of the NFL teams sorted by team name.
4. Display the list of NFL stadiums and their corresponding team name sorted by stadium name.
5. Display the list of only the American Football Conference teams sorted by team name. The National Football Conference teams should not be displayed.
6. Display the list of only the National Football Conference teams sorted by team name. The American Football Conference teams should not be displayed.
7. Display the list of only the NFC North teams sorted by team name. No other teams should be displayed.
8. Display the list of only the stadiums that have an “open” stadium roof type and their corresponding team name sorted by stadium name. Display the number of stadiums have an “open” roof. If two teams use the same stadium, count the “open” roof just once.
9. Display the list of NFL star players and their corresponding team name sorted by team name.
10. Display the list of NFL stadiums and their corresponding team names sorted by seating capacity (smallest to largest). Be sure to display seating capacity. Display the total capacity of all NFL teams. If two teams use the same stadium, count the capacity of the stadium only once.
11. Display the list of NFL teams, their stadium names, their conference, and their corresponding location sorted by conference name.
12. Display all the souvenir list related to only one particular football team.

**Planning a vacation:**

1. Provide the capability for a football fan to visit any other team of their choice starting at the Green Bay Packers traveling the shortest distance. Your Agile team should implement *Dijkstra’s* or the A\* algorithm. Display the total distance travelled.
2. Provide the capability for a football fan to plan his/her dream vacation by allowing a football fan to choose their starting team and all the other teams they would like to visit using the order specified. Display the total distance travelled.
3. Provide the capability to visit the all the teams starting at New England Patriots traveling the shortest distance. Chose the team closest to the New England Patriots and then chose the team closes to that team, etc.
   1. Display the total distance traveled
4. Provide the capability for a football fan to plan his/her dream vacation by allowing a football fan to choose their starting team
   1. Then allow a football fan to select other teams they wish to visit
   2. Plan the trip starting with the selected team then visit each of the other teams in the most efficient order (recursively choose the team closest to the previous team)
   3. Display the total distance traveled
5. Determine the minimum spanning tree (MST) connecting all the NFL stadiums using Prim’s or Kruskal’s algorithm. Display the associated mileage.
6. Perform a DFS starting at the Minnesota Vikings. If there is a choice, always choose the shortest distance. Display the associated mileage.
7. Perform a BFS starting at Los Angeles Rams. If there is a choice, always choose the shortest distance. Display the associated mileage.
8. When taking any trip:
   1. A football fan can purchase multiple souvenirs
   2. Your Agile team must keep track of the number of souvenirs purchased at each stadium.
   3. Display the total amount spent at each stadium and a grand total for all stadiums visited
9. Maintenance (administrator only - requires a password to gain access)
   1. Provide the capability to add a new team and its corresponding information by having your program read from an input file given to the class (assume the team has the default souvenirs)
   2. Provide the capability to change the prices of the traditional souvenirs
   3. Provide the capability to add new traditional souvenirs
   4. Provide the capability to delete traditional souvenirs
10. Provide the ability to modify stadium information including capacity if a team moves into a new stadium. (administrator only) (The Buffalo Bills are planning to move to Bills Field in 2026 with a new capacity of 60,000).
11. What is the total seating capacity of the NFL teams?
12. List and display the count of the number of NFL teams whose surface type is Bermuda Grass.

The initial souvenir list is follows:

1. Signed helmets $74.99
2. Autographed Football $79.89
3. Team pennant $17.99
4. Team picture $29.99
5. Team jersey $199.99

(Each team needs its own souvenir list so it can be modified)

Please let me know your partners by October 24h (three points will be deducted from your scores if you do not meet this deadline). All projects are due by December 7th. **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
   1. Three use cases
   2. One activity diagram
   3. Three state diagrams with your project.
5. Submit a test plan.
6. All changes must be persistent between executions.
7. Submit a **Big-Oh** discussion 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 – November 9th – 4 points

Second checkpoint – November 23nd – 4 points

Final checkpoint – December 7th - 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
6. Code, test plan, agile stories, scrum log, coding standards, team rules, UML diagrams (class, use cases, state diagrams), data structures used, Big Oh analysis