Team Durian’s Stories:

1. **Description**: As a baseball fan, I would like the program to display all the information related to only one particular baseball team. For example, the program will display the team's stadium name, seating capacity, location, playing surface, team name, league, date opened, distance to center field, ballpark typology, and roof type so that they can understand more about one specific team.
   Assumptions: A database is created and allows for retrieval of all the items necessary for one specific team.
   Assignee: Mark
   Estimate: 4
   Priority: Sprint One
   List of Task: Create a function that will display one team at a time. When displaying each team, the program must also display the team's stadium name, seating capacity, location, playing surface, team name, league, date opened, distance to center field, ballpark typology, and roof type.
   List of Test: To test the code, run the program and then, as the user, pick any team. When the team is picked then, have the program output all the information about that team and only that team.
   Def. of Done: As a user of this application, I am able to select any college, and the program will then output all the necessary information about that team. Each team has their own information, and the program only displays the selected team's information.

2. **Description**: As a fan I would like to see a list of MLB stadiums that are sorted by the team’s name alphabetically so that I know what my favorite team’s stadium is called.
   Assumptions: This will be sorted by team name and will contain a stadium for each team.
   Assignee: Julien
   Estimate: 6
   Priority: Sprint One
   List of Task: Create a function that will display a list of all the major league teams as well as their stadium names and have the program sort the list by team names. This will require the assignee to create a sort function to sort the list before displaying it to the user.
   List of Test: To test the program, a user asks for the program to display a list of all the major league teams and their corresponding stadiums. The program will then display a list sorted by team names.
   Def. of Done: As a user of this application, I am able to ask the program to display all the major league teams as well as their stadiums in a sorted list based on their team names.
   // 2 is already sorted so there's really no need for this but it is important for the fan.

3. **Description**: As a baseball fan, I would like to see a list of teams and stadiums sorted by stadium name alphabetically.
   Assumptions: There will be a search by stadium or search by name in the GUI. This will be a stadium.
   Assignee: Julien
   Estimate: 6
   Priority: Sprint One
   List of Task: Create a function that will display a list of all the major league teams as well as their stadium names and have the program sort the list by stadium names. This will require the assignee to create a sort function to sort the list before displaying it to the user.

List of Test: To test the program, a user asks for the program to display a list of all the major league teams and their corresponding stadiums. The program will then display a list sorted by stadium names.
Def. of Done: As a user of this application, I am able to ask the program to display all the major league teams as well as their stadiums in a sorted list based on their stadium names.

4. **Description**: As a baseball fan, I would like the program to display the list of American League team names and their corresponding stadium names sorted by team name.
   Assumptions: We can sort out American from National League.In the GUI there will be a selection to search all MLB, AL, or NL teams.
   Assignee: Julien
   Estimate: 6
   Priority: Sprint One
   List of Task: Create a function that will display a list of all the American league teams as well as their stadium names and have the program sort the list by team names. This will require the assignee to create a sort function to sort the list before displaying it to the user.
   List of Test: To test the program, a user asks for the program to display a list of all the American league teams and their corresponding stadiums. The program will then display a list sorted by team names.
   Def. of Done: As a user of this application, I am able to ask the program to display all the American league teams as well as their stadiums in a sorted list based on their team names.

5. **Story:** As a baseball fan, I can view a list of the National League teams and their stadiums sorted alphabetically by the stadium names
   Description: Display the list of National League team names and their corresponding stadiums names sorted by stadium name.
   Assumptions:
   - The database for the list of national league teams and stadiums has been initialized
   - GUI is created

   Assignee: Trevor
   Estimate: 6
   Priority: Sprint One
   List of Task:
   - Create a function that will only display teams from the National League
   - Display the stadiums associated with the baseball teams
   - Sort the list alphabetically by the stadium names using a sorting algorithm

   List of Test:
   - Ensure only teams from the national league are displayed
   - Ensure the list is in order by the stadium's name
   - Ensure the correct stadiums are associated with its correct team

   Def. of Done:
   - The fan can view the list of National League Teams and their corresponding stadiums sorted by the team names

6. **Story:** As a fan, I want to see the teams and their parks sorted by park typology.
   Description: Display the list of stadiums and their corresponding team name sorted by park typology. Be sure to display park typology.
   Assumptions:

- The database contains the list of team names and park typologies
- GUI is created

Assignee: Trevor
Estimate: 6
Priority:Sprint One
List of Task:
6 different type of typologies.
Drop down for : retro modern, retro classic, jewel box, modern, contemporary, and multipurpose

- Create a function that will only team names
- Display the park typologies associated with the the baseball teams
- Sort the list alphabetically by the park typology

List of Test:

- Ensure all team names are displayed
- Ensure the list is in order by the park typology
- Ensure the correct park typologies are associated with its correct team

Def. of Done:

- The fan can see the baseball team names and their corresponding park typology.

7. **Story:** As a fan, I want to see a list of teams that correspond with roof types
Description: Display the list of team names that have an open roof type sorted by team name. Display the number of teams with an open roof type.
Assumptions:
    - The database contains the list of team names and the list of stadiums with open roofs
    - GUI is created
    - Fixed is closed
    - Retractable means you can open or close the roof.
    - Drop down like story 6
    - Open is open

    Assignee: Trevor
    Estimate: 5
    Priority:Sprint One
    List of Task:
    - Create a function that will only team names
    - Display the roof types associated with the the baseball teams
    - Sort the list alphabetically by the team name

    List of Test:
    - Ensure all team names are displayed
    - Ensure the list is in order by the baseball team’s name
    - Ensure the roof types are associated with their corresponding team.

    Def. of Done:
    - The fan can view the list of teams and their corresponding roof types

8. **Story:** As a fan, I want to view a list of stadiums and their teams in chronological order of date opened so that I can find an old or new or something between the stadiums I prefer.
Description: Display the list of stadiums and their corresponding team name in chronological order (oldest to newest) using date opened. Be sure to display the date opened.
Assumptions:
    - The database contains the list of team names and the dates the teams were established
    - GUI is created

Assignee: Julien
Estimate: 6
Priority:
List of Task:

- Create a function that displays team names
- Display the corresponding stadium with the baseball teams
- Display the corresponding date opened for the stadiums
- Sort the list by the date opened from oldest to newest

List of Test:

- Ensure all the teams are displayed on the list
- Ensure the correct stadium is associated with their team
- Ensure the correct date is displayed with their teams and stadium
- Ensure the list is in order from oldest to newest

Def. of Done: The fan is able to view the list of stadiums and their associated team names in chronological order of date opened.

9. **Story:** As a fan, I would like to be able to sort stadiums by seating capacity so that I can find a stadium I want to visit more efficiently.
Description: This function is meant to create a sorted list that, when called, displays a list of stadiums. Stadiums that have the highest seating capacity are at the top of the list, while stadiums with the lowest seating capacity are at the bottom.
Assumptions: The data is stored in a database and is easily accessible. There is a sort function that will order the stadium list from highest seating capacity to lowest seating capacity.
Assignee: Yaqub
Estimate: 4
Priority:Sprint One

List of Task: To complete this task, the programmer must retrieve the necessary information from the database. After that, the programmer must use the newly created sorting function to sort the information and then display it in a list.

List of Test:To test this function, the programmer can run the program and then click the function's assigned button to see if the list outputted is correct.

Def. of Done:The function is successfully completed when it returns an ordered list with the stadiums that have the greatest seating capacity at the top and the stadiums with the lowest seating capacity at the bottom.

10. **Story:** As a fan, I would like to be able to sort stadiums by distance to center field by smallest to largest so that I can find a stadium I want to visit more efficiently.
Description: This function is meant to create a sorted list that, when called, displays a list of stadiums. Stadiums with the shortest distance to center field are at the top of the list, while stadiums with the greatest distance to center field are at the bottom.
Assumptions:The data is stored in a database and is easily accessible. There is a sort function that will order the stadium list from stadiums with the shortest distance to center field to stadiums with the greatest distance to center field.
Assignee: Yaqub
Estimate: 5
Priority:Sprint One

List of Task: To complete this task, the programmer must retrieve the necessary information from the database. After that, the programmer must use the newly created sorting function to sort the information by stadiums with the shortest distance to center field at the top and stadiums with the greatest distance to center field at the bottom, and then display the information in a list.

List of Test: To test this function, the programmer can run the program and then click the function's assigned button to see if the list outputted is correct.

Def. of Done: This function is finished when it returns an ordered list of stadiums with the smallest distance to center field at the top and the greatest distance to center field at the bottom.

11. **Story:** As a fan, I would like to be able to sort stadiums by distance to center field by largest to smallest so that I can find a stadium I want to visit more efficiently.
    Description: This function is meant to create a sorted list that, when called, displays a list of stadiums. Stadiums with the greatest distance to center field are at the top of the list, while stadiums with the shortest distance to center field are at the bottom.
    Assumptions: The data is stored in a database and is easily accessible. There is a sort function that will order the stadium list from stadiums with the greatest distance to center field to stadiums with the shortest distance to center field.
    Assignee: Yaqub
    Estimate: 5
    Priority:Sprint One

List of Task: To complete this task, the programmer must retrieve the necessary information from the database. After that, the programmer must use the newly created sorting function to sort the information by stadiums with the greatest distance to center field at the top and stadiums with the smallest distance to center field at the bottom, and then display the information in a list.

List of Test: To test this function, the programmer can run the program and then click the function's assigned button to see if the list outputted is correct.

Def. of Done: This function is finished when it returns an ordered list of stadiums with the greatest distance to center field at the top and the smallest distance to center field at the bottom.

12. **Description**: Provide the capability for a baseball fan to visit any other team of their choice starting at the Dodger Stadium (Los Angeles Dodger) traveling the shortest distance. Your Agile team should implement *Dijkstra's* or the A* algorithm. Display the total distance traveled.
    Assumptions: The teams are already sorted and easily accessible in a database or another data structure in order to determine which order to visit each of the teams that the baseball fan wants to visit.
    Assignee: Trevor
    Estimate: 9
    Priority: Sprint Two
    List of Task: Use Dijkstra's Algorithm to determine the most efficient trip for the baseball fan's choice of teams.
    List of Test: Plan a custom trip and choose four different teams to visit and check the output trip travel and thet total distance. Then check the
    Def. of Done: Goes through multiple trips with different baseball teams and accurately outputs the most efficient trip with the correct distance travel.

13. **Description**: Provide the capability for a baseball fan to plan his/her dream vacation by allowing a baseball fan to choose their starting team and all the other teams they would like to visit using the order specified using the shortest path. Display the total distance traveled.
    Assumptions: The teams are already sorted and easily accessible in a database or another data structure in order to determine which order to visit each of the teams that the baseball fan wants to visit.
    Assignee: Trevor
    Estimate: 8

Priority: Sprint Two

List of Task: Allow the user to choose any stadium they want to, and calculate the most efficient trip from the baseball fan's choices.

List of Test: Choose random stadiums and check with the database and values to see that the distance traveled is correct and that they order the program visited the stadiums in is the most efficient.

Def. of Done: The baseball fan/user can choose any stadium in any order that they would like to visit and take them on the most efficient trip possible and show the order of the stadiums the program visited.

14. **Description**: Provide the capability to visit all the teams starting at Marlins Park (Miami Marlins) traveling the shortest distance. Choose the team's stadium closest to Marlins Park and then choose the stadium closest to that stadium, etc. Display the total distance traveled.

    Assumptions: The teams are already sorted and easily accessible in a database or another data structure in order to determine which order to visit each of the teams that the baseball fan wants to visit.

    Assignee: Trevor

    Estimate: 7

    Priority: Sprint Two

    List of Task: Start from Marlins Park and visit all the teams from there and calculate the total distance traveled and make it the most efficient trip possible.

    List of Test: Start from Marlins Park and check to see the total distance is correct for all of the stadiums.

    Def. of Done: The trip outputs the correct distance and shows each of the stadiums it visited in the correct order.

15. **Description**: Provide the capability for a baseball fan to plan his/her dream vacation by allowing a baseball fan to choose their starting team. Then allow a baseball fan to select other teams they wish to visit. Plan the trip starting with the selected team's stadium then visit each of the other teams' stadium in the most efficient order (recursively choose the team closest to the previous team). Display the total distance traveled.

    Assumptions: The stadiums are already sorted and easily accessible in a database or another data structure in order to determine which order to visit each of the teams that the baseball fan wants to visit.

    Assignee: Trevor

    Estimate: 7

    Priority: Sprint Two

    List of Task: Let the baseball fan create a custom trip allowing them to visit any of the baseball teams in any order they want. When they take the trip, they must visit each stadium in the most efficient order.

    List of Test: Go from Marlins Park to Angels Stadium to another stadium and check if the distance is correct. In addition, check to see that the order that the program visited is the correct order.

    Def. of Done: The custom trip outputs the correct distance and takes the trip in the most efficient order.

16. **Story:** Perform a DFS starting at Oracle Park (San Francisco Giants). If there is a choice, always choose the shortest distance. Display the associated mileage.

Description: Use a DFS to go through all of the stadiums in the database and choose the shortest distance.

Assumptions: The database is already set up, the code for the DFS has already been worked on through previous assignments and is proven to work on other data and examples.

Assignee:Trevor

Estimate: 8

Priority: Sprint Two

List of Task: Let the baseball fan start the trip at Oracle Park, and have the program perform a DFS and show that the DFS works with all of the stadiums.

List of Test: Compare the results of the DFS of the trip to the results from the DFS previously worked on in the homework to check if the DFS algorithm is working properly.

Def. of Done: As a user, they will be able to choose the DFS trip. Starting from Oracle Park, they will go through all of the stadiums and it will show the exact route the DFS algorithm chose to take to show that the DFS was successfully completed.

17. **Story:** Perform a BFS starting at Target Field (Minnesota Twins). If there is a choice, always choose the shortest distance. Display the associated mileage.

    Description: Use a BFS to go through all of the stadiums in the database and choose the shortest distance.

    Assumptions: The database is already set up, the code for the BFS has already been worked on through previous assignments and is proven to work on other data and examples.

    Assignee:Trevor

    Estimate: 8

    Priority: Sprint Two

    List of Task: Let the baseball fan start the trip at Target Field, and have the program perform a BFS and show that the BFS works with all of the stadiums.

    List of Test: Compare the results of the BFS of the trip to the results from the BFS previously worked on in the homework to check if the BFS algorithm is working properly.

    Def. of Done: As a user, they will be able to choose the BFS trip. Starting from Target Field, they will go through all of the stadiums and it will show the exact route the BFS algorithm chose to take to show that the BFS was successfully completed.

18. **Story:** admin for souvenirs

    Description: As an admin, I would like to add or remove traditional souvenirs and change its price for each stadium

    Assumptions: There will be a functionality and ui button on admin page to add or remove souvenirs and change pricing

    Assignee: Mark

    Estimate: 5

    Priority: Sprint Two

    List of Task: Create a function and ui button to add or remove a souvenir to the database with corresponding stadium. Create another function to change the price of a corresponding souvenir in the database. This function needs access to the database.

    List of Test: To test the program, an admin adds and removes a souvenir in the maintenance page, and checks if the item is added/removed in the database. Same thing for changing the price.

    Def. of Done: As an admin, I am able to ask the program to add/remove souvenirs and change its corresponding price on the maintenance page.

19. **Story:** Admin for baseball teams.

Description: As an admin, I am able to add a new stadium and its corresponding souvenirs by reading an input file
Assumptions: There will be a maintenance ui to allow inputting file and submitting data to database
Assignee:Mark
Estimate: 6
Priority: Sprint Two
List of Task: Create a function(s) and ui buttons to read input file (excel), and push the new stadium and souvenirs to the database.
List of Test: To test the program, an admin input the file to the program and check if the data is added to the database
Def. of Done: As an admin, I am able to ask the program to add new stadiums and its corresponding souvenirs by inputting a excel sheet file

20. **Story:** admin for stadium attributes like seating capacity, location, etc.
Description: As an admin, I am able to modify all the related information of a stadium for a team.
Assumptions: There will be a maintenance ui to allow user change all the information for a team
Assignee: Mark
Estimate: 6
Priority: Sprint Two
List of Task: Create a function to modify the following information in the database for each team: capacity, stadium name, playing surface, roof type, ballpark typology, date opened, distance to center field, and location if a team moves into a new stadium.
List of Test: To test the program, an admin change every stadium information for a team and check if the data is modified in the database
Def. of Done: As an admin, I am able to change any stadium information regarding a team.

21. **Story:** as an admin, I want to log in to the program with a encrypted password
Description: As an admin, I want to log in to the program with a encrypted password
Assumptions: N/A
Assignee:Mark
Estimate: 1 (BASELINE)
Priority: 1
List of Task: Create a function and UI of a login page that requires admin to input accurate username and password to sign in. The password needs to be encrypted.
List of Test: To test the program, an admin input the username and password to test if the program successfully logs in.
Def. of Done: As an admin, I am able to sign in into the maintenance page by inputting the correct user name and password on the login page.

22. **Story:** Create a souvenir stand for each time you visit a stadium
Description: As an admin, I want to log in to the program with a encrypted password
Assumptions: N/A
Assignee:Mark
Estimate: 1 (BASELINE)
Priority: 1
List of Task: Create a function and UI of a login page that requires admin to input accurate username and password to sign in. The password needs to be encrypted.
List of Test: To test the program, an admin input the username and password to test if the

program successfully logs in.
Def. of Done: As an admin, I am able to sign in into the maintenance page by inputting the correct user name and password on the login page.

Team Rules:

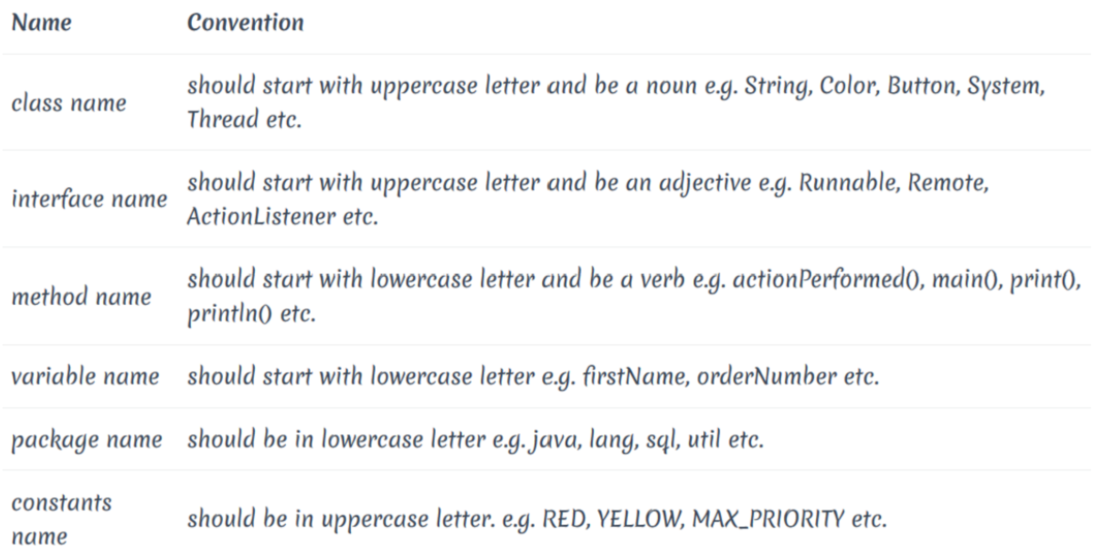
- Respect your group members
- Use coding standards when coding.
- Before pushing code to Github follow Github process
- If you are unable to do your assigned work message the group
- Leave comments on your code.
- Don't be afraid to ask for help
- Have someone check your code before pushing
- using When2meet to set the actual time to meet in each week
- We need to meet 1-2 every week outside of class (Tuesday- Sunday)

GitHub process:

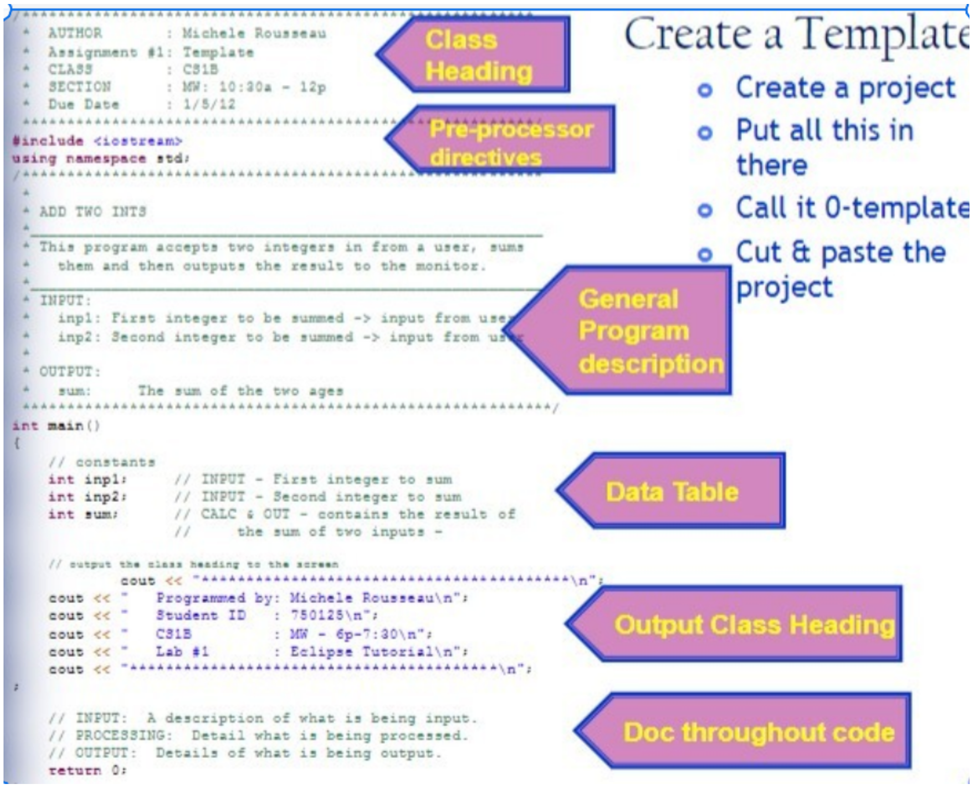
- Pull latest version from the GH:
- Make sure you are on the main branch
- git pull --rebase
- git checkout -b <branch_name>
- git checkout main
- git pull --rebase
- git checkout <your feature branch>
- git merge main
- git push

Coding Standards:

- Variable naming conventions (Camel case)
- Class and function naming conventions
- Clear and concise comments
- Indentations
- Portability
- Reusability and scalability
- Testing



Name	Convention
class name	should start with uppercase letter and be a noun e.g. String, Color, Button, System, Thread etc.
interface name	should start with uppercase letter and be an adjective e.g. Runnable, Remote, ActionListener etc.
method name	should start with lowercase letter and be a verb e.g. actionPerformed(), main(), print(), println() etc.
variable name	should start with lowercase letter e.g. firstName, orderNumber etc.
package name	should be in lowercase letter e.g. java, lang, sql, util etc.
constants name	should be in uppercase letter. e.g. RED, YELLOW, MAX_PRIORITY etc.



Create a Template

- Create a project
- Put all this in there
- Call it 0-template
- Cut & paste the project

```
/************************************************************
 *  AUTHOR        : Michele Rousseau
 *  Assignment #1: Template
 *  CLASS         : CS1B
 *  SECTION       : MW: 10:30a - 12p
 *  Due Date      : 1/5/12
 *************************************************************/
#include <iostream>
using namespace std;
/************************************************************
 *
 * ADD TWO INTS
 *____________________________________________________________
 * This program accepts two integers in from a user, sums
 *   them and then outputs the result to the monitor.
 *____________________________________________________________
 * INPUT:
 *   inp1: First integer to be summed -> input from user
 *   inp2: Second integer to be summed -> input from user
 *
 * OUTPUT:
 *   sum:     The sum of the two ages
 *************************************************************/
int main()
{
    // constants
    int inp1;       // INPUT - First integer to sum
    int inp2;       // INPUT - Second integer to sum
    int sum;        // CALC & OUT - contains the result of
                    //      the sum of two inputs -

    // output the class heading to the screen
            cout << "*****************************************\n";
    cout << "   Programmed by: Michele Rousseau\n";
    cout << "   Student ID    : 750125\n";
    cout << "   CS1B          : MW - 6p-7:30\n";
    cout << "   Lab #1        : Eclipse Tutorial\n";
    cout << "*****************************************\n";
;

    // INPUT:  A description of what is being input.
    // PROCESSING:  Detail what is being processed.
    // OUTPUT:  Details of what is being output.
    return 0;
```

Class Heading

Pre-processor directives

General Program description

Data Table

Output Class Heading

Doc throughout code