Ivan Ramos
Owen Bratt
Javi Buenrostro
Hetgar Peralta
Kurtis Chan

CSC 340 Group 4 Project

a.A list of algorithms described in pseudocode with necessary comments

```
readFromCSV:
string teamString;
team_data currentTeam;
put a line of the csv into teamString //overridden in next line to get rid of title line
put a line of the csv into teamString //prepare for start of loop
for(int teamNum = 1; file remains open; teamNum++)
    make a team object from teamString //call helper function, createTeam()
    put the team object in a list
    put a line of the csv into teamString

getMeanAveDefence()
  //for loop to get the data from the team vector dividing it by the size
getTeamStatRating(string& TeamName);
  //for loop to get the data, if to check if TeamName is on the dataset then subtracts the collected offensive from the defensive, printing the Team Name's stat and returning the Stat
teamImproved()
  For each team, find the winDiff, adjODiff, adjDDiff, and rankDiff of both years and if the value is greater than 0, then the team has improved
```

b. A list of built-in datatypes you have selected to use with a brief justification (*no need to include simple data types such as int and char)*

For built-in datatypes, we've decided to use vectors, as we're familiar with the way they work, and it suits our needs well.

c. A list of user-defined datatypes with a brief justification (no need to include the entire header file).

For user-defined datatypes, we've created a class named 'teamData', which stores information about each individual team. This includes the team name, rank, games played/won, seed, etc…, and the class also includes getter and setter files, as well as a default constructor.
The other class, 'yearlyTeamData', contains the private vector 'yearlyData', which contains teamData objects. This class has access to most of our user-defined functions, such as 'getMeanAveDefence', 'readFromCSV', etc… The constructor for this class actually uses the 'readFromCSV' function in order to push values into the vector of the yearlyTeamData object, depending on the file you give it.

d. A summary of the strategies adopted by your team to guarantee that your programs produce the correct results (e.g., unit tests and integration tests, peer code review.)

Most of the testing was done in the main file, Javi had Hetgar check the code and they ran tests while coding that portion

e. A summary of the strategies adopted by your team to improve the efficiency (in time and space) of your implementation

Originally, some of the functions were passing vectors by value, so Ivan modified these to pass them by reference instead.

f. Briefly reflect on the following two questions:

- What have you learned from this project individually and as a team?
    - Owen - Individually, I learned how much easier it is to think of large projects in a modular way when the tasks are actually divided among different people.
    - Kurtis - Just as Owen said, the did seem like a large task at first, but once the problem was broken down and tasks were divided between the five of us, it seemed doable.
    - Ivan - Seeing as I was in charge of the classes, I needed to make sure that the functions were modular, and that they worked as intended. For the second class, ‘yearlyTeamData’, I had to learn how to piece together different individual functions into a class, and modify them in ways so that they can work off each other. As the others said, when the work is divided, and each person has their own role, the assignment becomes much easier.
    - Javi - I learned how to use vectors and drag out specific data from the datafield and learned how to use github on a group level.
    - Hetgar - I say the important thing I learned from this project was the experience of working as a group. Also, I learned how to link the input files from another part of the file to the functions we worked on. Wish I knew how to do that before coding lab # 4.
- What were the major obstacles and how did you overcome them individually or as a team?

    Ivan - When testing the teamImproved function, I noticed I could not pass vectors from the yearlyTeamData objects into the function using the getter function alone. It had something to do with the fact that the teamImproved function’s vector parameters were passed by reference, because when I changed it to pass by value, the function would work. I found a workaround to this by first creating a testing vector in the main function, setting those to the vectors of the yearlyTeamData objects, and then passing those testing vectors.

    Javi - A major obstacle would be the CmakeList (automated building processor) not being at the correct format thus hindering the build and run process (plus costing having to restart the thing from scratch) me and Hetgar decided to keep the important codes on a notepad and delete the whole thing just to get the CMakeList to show up as a automatic process while annoying it did allow for the program to be run after doing it so I'd say it was overcomed.

Hetgar - As for the second question, the major obstacle was the CMakeList.txt. Near the end of our code, me and Javi encounter a problem where the output would result in nothing. Which was strange since beforehand everything would run perfectly. After multiple trials, we overcame this problem by restarting the project. We copied and pasted the code (that we thought worked) on a notepad and pasted it to a new project file. We then clicked the create CMakelist and everything worked/ran again.

Owen - One obstacle was that when pushing the code for csv_reader.cpp via git, it did not change the CMake application's working directory, and so the file path was not parsed properly. To fix this, Ivan, who was testing it, had to specify the working directory.

Kurtis - Since we changed one of the problems, the hardest part was coming up with a new one that would be complex enough to not be a simple search or sort, but also be doable within the short time we had. Writing the code wasn't the worst part of it and debugging was also not too bad of a task. The thought of making all these programs work together worried me, but it worked out in the end.

G. *A summary of each teammate's contribution

- Javi and Hetgar worked on the 2nd problem that included the getMeanAveDefence() and getTeamStatRating() functions
- Kurtis worked on the teamImproved() function which is problem 1
- Ivan worked on management of the classes and testing of functions
- Owen worked on the readFromCSV() function to package the data from the csv files into teamData classes.
- Finally, We all worked on the reflection together.