# CSC-24400 Programming Project #2
**DUE: Thursday, March 18, 2021, 11:59 PM**

## 1 Objectives

- writing a C++ class on your own
- working with dynamic partially filled array
- using the const keyword relative to parameters, methods and return values

## 2 Problem Statement

You will be writing a `C++` class called `SoccerLeague` that represents a league (i.e. set) of Soccer Teams. You are being given a class called `SoccerTeam` - both definition (`SOccerTeam.hpp`) and implementation (`SoccerTeam.cpp`); please make sure to use this class, as it should make your task *much* easier.

You *must* store the definition for your class (which *must* be called `SoccerLeague`) in a file called `SoccerLeague.hpp`. You may place your implementation in any `.cpp` file you like (although I'd recommend `SoccerLeague.cpp` for sanity's sake.) Your `SoccerLeague` class must contain the following public methods:

- a default constructor that takes no arguments. This should simply make the league be capable of holding up to 5 teams, but initially holds zero teams.
- a constructor that takes a single argument of type `iftsream` . This constructor should read Team data from the input stream, adding as many teams as there are in the input stream to the collection. Each team in the input stream will be specified on 3 different lines, as follows:

  ```
  <team-name>
  <wins>
  <losses>
  ```

  Where:

  - `<team-name>` is a string representing the name of a team. Note that team names *can* have spaces in them, but will not have newlines in them.
  - `<wins>` is an integer representing the number of game this team has won.
  - `<losses>` is an integer representing the number of game this team has lost.

  Several example data files meeting this format are being provided to you.
- a copy constructor that takes in as its only parameter an existing `SoccerLeague` and duplicates the existing `SoccerLeague`. The parameter should not be allowed to change and should be passed efficiently.
- the `+` operator should be overloaded. The right hand side of this operator should be another `SoccerLEague` (which should be guaranteed not to change and should be passed efficiently.) Using this operator should not cause either of the operands to change. The method should efficiently return the "union" of the two leagues. That is, each team from either operand's collection should be in the collection being returned but there should be no duplicates (i.e. no two teams with the same name.) If a team is found in both leagues, add the number of wins (and losses) together to get the new number of wins (or losees, respectively.)

- the `/` operator should be overloaded. The right hand side of this operator should be another `SoccerLeague` (which should be guaranteed not to change and should be passed efficiently.) Using this operator should not cause either of the operands to change. The method should efficiently return the "intersection" of the two collections. That is, all teams in the answer should be found in both operand's collections. For each team found in both leagues, add the number of wins (and losses) together to get the new number of wins (or losees, respectively.)
- the `=` operator should be overloaded. The right hand side of this operator should be another `SoccerLeague` (which should be guaranteed not to change and should be passed efficiently.) Using this operator should cause the operand on the left hand side to change to be a copy of the one on the right hand side. Be warned that the result should not "share" the same internal array! Finally, this method should return the copied `SoccerLeague`.
- the `+=` operator should be overloaded with the right hand side being a single `SoccerTeam`. Make sure to pass the `SoccerTeam` efficiently and guarantee that it will not be modified by this method. This method should simply add the Team corresponding to the parameter to this league (making sure it is not already in the league.) Finally this method should return a copy of the new league as efficiently as possible.
- the `+=` operator should also be overloaded with the right hand side being another `SoccerLeague`. Make sure to pass the right hand side soccer league efficiently and guarantee that it will not be modified by this method. This method should simply add all of the the teams corresponding to the parameter to this league (making sure each team is not already in the league.) Finally this method should return a copy of the new league as efficiently as possible.
- the `-=` operator should be overloaded with the right hand side being a single `SoccerTeam`. Make sure to pass the `SoccerTeam` efficiently and guarantee that it will not be modified by this method. This method should simply remove the team corresponding to the parameter from this collection (or do nothing when the team is not in the current collection.) Finally this method should return a copy of the new league as efficiently as possible.
- the `-=` operator should also be overloaded with the right hand side being another `SoccerLeague`. Make sure to pass the right hand side league efficiently and guarantee that it will not be modified by this method. This method should simply remove each of the the teams in the collection corresponding to the parameter from this collection. Finally this method should return a copy of the new league as efficiently as possible.
- a method called `clear` that simply empties out the current league. After calling this, it should appear as if there are no teams in the current league.
- a method called `print` that should take a modifiable `ostream` object as its only parameter. This method should return a modifiable reference to an `ostream` object. Essentially, the method should just print out every team in the current league to the stream being passed in to the method, one movie per line. Lastly, this method should return the updated output stream.
- the `<<` operator should be overloaded for a `SoccerLeague` object. Remember that this method should return a reference to an object of type `ostream`.

A few final notes:

- You *MUST* use a partially filled array to represent your `SoccerLeague`. Using anything else will result in a *very* poor grade on this project.
- Whenever you need more space than your partially filled array currently has, you should allocate a new array of twice the size of the current one.

## 3 What To Hand In

You will be submitting a zip or tgz file containing your source code and a `read.me` file to Canvas. Make sure that you place the project into a single folder (which may contain sub-folders).

The `read.me` file should include information about your project including (but not limited to):

- your name
- the date
- the platform you developed your code on (Windows, Linux, ...)
- any special steps needed to compile your project
- any bugs your program has
- a brief summary of how you approached the problem

You might also want to consider adding things like a "software engineering log" or anything else you utilized while completing the project.

## 4 Grading Breakdown

| | |
|---|---|
| Correct Submission | 10% |
| Code Compiles | 15% |
| Following Directions | 30% |
| Correct Execution | 30% |
| Code Formatting/Comments/`read.me` | 15% |
| *Early Submission Bonus* | *5%* |

## 5 Notes & Warnings

- For most people, this is not the kind of project that you will be able to start the day before it is due and successfully complete. My recommendation is to start *now*.
- If you have any questions about anything involved with this project, ask me. If you don't ask for help, I will not know that you want it. Do not hesitate to attend my virtual office hours or e-mail me questions!
- *Start now*!
- Projects may *not* be worked on in groups or be copied (either in whole or in part) from *anyone or anywhere*. This also means that you may not allow your code to be copied, either in part or in whole, by anyone - such would mean that you (as a provider of code) have also cheated. Failure to abide by this policy will result in disciplinary action(s). See the course syllabus for details.
- *START NOW*!
- Have you started working on this project yet? If not, then *START NOW !!!!!*