CS 201, Fall 2015
Homework Assignment 1

In this homework, you will implement a soccer tournament team registration system (SocReg). There are going to be multiple teams registering for the tournament. For each team you are going to store a record.

Each team has a name, roster, and color.

In your implementation, you **MUST** keep the teams in a dynamically allocated array. For each of these teams, you **MUST** use another dynamically allocated array to keep its players. Neither the number of teams nor the number of players in a team is known in advance.

Your system will have the following functionalities; the details of these functionalities are given below:

1. Add a team
2. Remove a team
3. Add a player to the team
4. Remove a player from the team
5. Display all registered teams
6. Show detailed information about a particular team
7. Find which team(s) a player is playing for

Add a team: This function adds an entry to the system for a given team whose name and color are specified as parameters. In this function, the player list is not specified; they are going to be added later. Names of the teams are unique. If user attempts to enter a team with an already registered name, they should be warned and no records should be entered. Teams can have the same color.

Remove a team: This function removes a team entry, whose name is specified as a parameter. If there are no teams with the given name in the system, user should be warned and the list of teams should not be modified. Note that this function also removes the player list of the team.

Add a player to the team: This function adds a player to the roster of a team. Team name the player is playing for, player name, position of the player (e.g., forward, defense, midfield, goalkeeper) are specified as parameters. Your function should take care of the following issues (1) If the team with a specified name does not exist in the system, display a warning message and do not add that player. (2) In this system, all player names are unique within the same team. Thus, if the user attempts to add a player with an existing name in the same team, print a warning message and do not add that player. However, different teams can have players who share the same name.

Remove a player from the team: This function removes a player from the roster of a team. Team name the player is playing for and player name are given as parameters. If there are no teams with a specified name, or if the specified player name is not in the roster of the specified team, display a warning message and do not perform the requested action.

Display all teams: This function lists all teams already registered in the system, in the following format. If the there are no teams, display `--EMPTY--`.

```
Team Name, Color (for the 1st team)
Team Name, Color (for the 2nd team)
. . .
```

Show detailed information about a particular team: This function displays all of the information about a team whose name is specified as a parameter. The output should be in the following format. If the team with a specified name does not exist in the system, you should display `--EMPTY--` after the first line.

```
Team, color
Player name, position (for the 1st player)
Player name, position (for the 2nd player)
. . .
```

Find the team(s) a player is playing for: This function lists the team a player is playing for. As multiple teams can have players who have the same name, given a name you might have multiple teams associated with this player name. The name of the player is the only parameter. The output should be in the following format. Note that if the specified player does not participate in any roster, you should display `--EMPTY--` after the first line.

```
Player name
Position, team name, team color (for the 1st team)
Position, team name, team color (for the 2nd team)
. . .
```

Below is the required public part of the `SocReg` class that you must write in this assignment. The name of the class **must** be **`SocReg`**. The interface for the class must be written in a file called `SocReg.h` and its implementation must be written in a file called `SocReg.cpp`. Your class interface file ("`SocReg.h`") should contain the following member functions. If necessary, you may also define additional public and private member functions and data members in this class. You can also define additional classes in your solution. **On the other hand, you are not allowed to delete any of the given functions or modify yhe prototype of any of these given functions**.

```
#include <string>

class SocReg {
public:
     SocReg();
     ~SocReg();

     void addTeam( string teamName, string color );
     void removeTeam( string teamName );
     void addPlayer( string teamName, string playerName, string playerPosition );
     void removePlayer( string teamName, string playerName );

     void displayAllTeams();
     void displayPlayer( string playerName );
     void displayTeam( string teamName );

     // ...
     // you may define additional member functions and data members,
     // if necessary.
};
```

NOTES ABOUT IMPLEMENTATION:

1) You ARE NOT ALLOWED to modify the given parts of the header file. You MUST use dynamically allocated arrays in your implementation. You will get no points if you use fixed-sized arrays, linked-lists or any other data structures such as `vector/array` from the standard library. However, if necessary, you may define additional data members and member functions.
2) Moreover, you ARE NOT ALLOWED to use any global variables or any global functions.
3) Your code must not have any memory leaks. You will lose points if you have memory leaks in your program even though the outputs of the operations are correct.
4) Your implementation should consider all team, color, and player names as case insensitive. For example, the team titles "FeNeRBaHCe" and "FENERBAHCE" should be considered as the same team.
5) Make sure that each file that you submit (each and every file in the archive) contains your name and student number at the top as comments.

NOTES ABOUT SUBMISSION:

This assignment is due by 23:59 on November 16, 2015. This homework will be graded by your TAs Necmi Acarsoy (necmi.acarsoy at bilkent edu tr) and Arda Unal (arda.unal at bilkent edu tr).

1. In this assignment, you must have separate interface and implementation files (i.e., separate .h and .cpp files) for your class. The file names should be "`SocReg.h`" and "`SocReg.cpp`". You should also submit other .h and .cpp files if you implement additional classes. We will test your implementation by writing our own main function. Thus, you should not submit any function that contains the main function.

   Although you are not going to submit it, we recommend you to write your own driver file to test each of your functions. However, you SHOULD NOT submit this test code (we will use our own test code).

2. You should put your "`SocReg.h`" and "`SocReg.cpp`" (and additional .h and .cpp files if you implement additional classes) into a folder and zip the folder (in this zip file, there should not be any file containing the main function). The name of this zip file should conform the following name convention: secX-Firstname-Lastname-StudentID.zip where X is your section number.

   The submissions that do not obey these rules will not be graded.

3. **Then, before 23:59 on November 16, you need to send an email with a subject line CS 201 HW1 to either Necmi Acarsoy or Arda Unal, by attaching this zipped file containing only your header and source codes (but not any file containing the main function).**

   No hardcopy submission is needed. The standard rules about late homework submissions apply. Please see the course syllabus for further discussion of the late homework policy as well as academic integrity.

4. You are free to write your programs in any environment (you may use either Linux or Windows). On the other hand, we will test your programs on "`dijkstra.ug.bcc.bilkent.edu.tr`" and we will expect your programs to compile and run on the `dijkstra` machine. If we could not get your program properly work on the `dijkstra` machine, you would lose a considerable amount of points. Therefore, we recommend you to make sure that your program compiles and properly works on "`dijkstra.ug.bcc.bilkent.edu.tr`" before submitting your assignment.
5. This homework will be graded by your TAs Necmi Acarsoy (necmi.acarsoy at bilkent edu tr) and Arda Unal (arda.unal at bilkent edu tr). Thus, you may ask your homework related questions directly to them.