CIS 246 – C++ Programming

Program:	**Week 10**
Points:	**20**
Chapter(s):	**10**
File(s) to Submit:	**Team.h, Team.cpp, Player.h, Player.cpp, Driver.cpp (as one zip file)** **Do not submit .dev files.**

Summary

Write a C++ application that creates a Team of player objects that could be used in a role-playing game. Print the players in a neatly formatted list.

This assignment builds on the homework for Week 8. Finish that program first, and this one will be easier.

In this assignment, you will practice:

- Overloading the compound assignment operator +=

There is extra credit available for this assignment.

Description

This program has exactly the same purpose as the program from week 08 – it will create Player objects, and then display them. The difference is in how this functionality will be implemented.

In this version of the program, the driver creates one object of the Team class. An array or vector to store Player objects will be located in the Team class.

The driver creates three objects of a Player class, adds them to the Team object via an overloaded operator, and then calls a print function of the Team class to display the Player information.

Requirements for the Player Class

Use the exact same Player class from the Week 8 program, separated into a header file and an implementation file.

Requirements for the Team Class

This class **must be separated into a header file (Team.h) and an implementation file (Team.cpp).**

The header file must contain an include guard.

private data members:

1. Array (using the array class template) or vector, using the Player class. Do not use the built-in C-style array.

public functions:

1. **Print** function – Use a **range-based for** loop to print the class and level of each Player object in the array or vector.
2. **Overloaded += operator function** – This function takes a **reference to a Player object** as an argument, and then adds it to the array or vector.

   It returns a **reference** to the Team object (itself). Team&

   Note – add the following as the last line of this function:

   return *this;

Extra Credit (Max 5 points)

In the Team class, overload the **prefix increment operator** to add a Player object that uses both **default arguments**. Use this operator in the driver when adding a **default Player object**:

Example: ++team;

Requirements for the Driver Program

Main function
The main function creates one Team object. Do not create an array or vector here!

The main function then creates the same three Player objects as before:

1. By prompting the user for both the class and the level
2. By prompting the user for just the class.
3. By using both default values

After instantiating each Player object, add it to the Team object using the **+= compound assignment operator**.

Example code:

```
Team team;
Player player1;      ← created via one of the 3 techniques from the Week 8 program.
team+=player1;       ← overloaded compound assignment operator +=
```

You may want to plan the logic for the overloaded += operator using a chart similar to the slides.

Once all three Player objects are added to the Team object, call the **print function of the Team object**, which will display the contents of the array or vector, as shown in the sample output.

Requirements for Full Credit on This Project

COMPLETE AND ACCURATE – Your program must compile, execute, and give accurate output.

FOLLOW ALL REQUIREMENTS ACCORDING TO THE INSTRUCTIONS – Follow the instructions as written for completing this project, even if you [think you] know a "better" way to do something.

COMMENTS – Include comments in your code. There must be a comment at the top of each source code or header file that includes your name, the assignment number, and a description of the code in that file. There must be comments at each important step in your algorithm that describes that step.

BEST PRACTICES – Follow best practices in C++ programming, including, but not limited to, appropriate use of private/public, appropriate use of classes and/or header files, sets & gets, white space, alignment, meaningful variable names, naming conventions, using statements, etc. Points will be deducted for sloppy code that is hard to read, even if it works, so pay attention to these details.

SUBMIT ALL FILES BEFORE THE DUE DATE – See the due date for this assignment on the course calendar in Canvas and review the submission requirements below.

ALL SUBMISSIONS MUST BE YOUR OWN WORK – Review the syllabus regarding plagiarism and the Joliet Junior College Academic Honor Code. Review the Get Help page in every module if you need help with this assignment.

Submission Requirements

All source code file submissions must be compressed into **ONE** Compressed Folder, or zip file. The dropbox for this assignment will only allow compressed files (.zip extension) to be uploaded.

Instructions for creating a Compressed Folder (zip file) are available here:
https://support.microsoft.com/en-us/help/14200/windows-compress-uncompress-zip-files

Do not submit your code in any other file format but .cpp and .h (within a zip). I will not accept links to online storage. You must submit the actual file. Do not submit executable files. Do not submit project files from an IDE.

Sample Run

Your output may vary depending on your default values. Note that there should be no difference in the output for the programs from week 8 and 10.

```
Enter player 1 class: Warrior
Enter player 1 level: 2

Enter player 2 class: Archer

Your team:
Player 1: Warrior        Level: 2
Player 2: Archer         Level: 0
Player 3: Knight         Level: 0
```