CIS 246 – C++ Programming

Program:	**Week 8**
Points:	**20**
Chapter(s):	**8, 9**
File(s) to Submit:	**Player.h, Player.cpp, Driver.cpp (as one zip file)**

Summary

Write a C++ application that creates player objects that could be used in a role-playing game. Each player object includes a class and a level. Print the players in a neatly formatted list.

In this assignment, you will practice:

- Separating the interface of a class from its implementation
- Using a constructor with default arguments

Description

The driver creates three objects of a Player class.

A Player object contains two data members: a class name and a level. (Note that class is used here in the context of a role-playing game.)

A Player object can be created in three different ways:

1. With both RPG class name and level specified by the user
2. With only the RPG class name specified by the user; the level is set to a default value
3. With both RPG class name and level set to default values

The driver prompts the user to create two of the Player objects. The third Player object is created with default values.

Once all three players are created, print each one in a neatly formatted chart.

Requirements for the Player Class

The Player class **must be separated into two files**: a header file (Player.h) and an implementation file (Player.cpp).

The header file must contain an include guard.

private data members:

1. string variable to hold the class of the player (Knight, Archer, Warrior, etc.)
2. integer variable to hold the level

public functions:

1. **Two-argument constructor with default values for both data members as arguments** The default values can be anything you think works. I used “Knight” and 0 for demonstration purposes, but you don’t have to use those specific values.
2. **Getter & setter functions** as needed

Requirements for the Driver Program

The main function prompts the user twice:

1. Prompt for both the class and the level
2. Prompt for just the class.

After each prompt, instantiate a Player object using the appropriate constructor.

The third Player object is instantiated using the default values.

Print each Player object’s information to the screen.

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 a 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.

```
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
```