**CIS 246 – Spring 2020**

|  |  |
| --- | --- |
| **Program:** | **7** |
| **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
* Using an array or vector
* Generating random integers

**Description**

The driver creates five objects of a Player class, adds them to an array or vector, and prints the contents of the array or vector.

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

The driver uses a random number to determine which method to use, creates the Player object appropriately and stores the Player object in an array or vector.

Once all five players are created, use a loop to 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.

1. **Getter & setter functions** as needed

**Requirements for the Driver Program**

1. **main**

The main function creates the array or vector using the Player class. You are free to use an array from the <array> class, a vector from the <vector> class, or a built-in array. Use a size of 5 if using an array.

Within a counter-controlled loop that executes five times:

* Generate a random number from a set of 3.
* Use a **selection statement** to check the random number and create a Player object in one of the three ways listed above.
* When the player object is created, add it to the array or vector. If you are using an array, you can assign the new object to the current element of the array. If you are using a vector, you can use the push\_back function.
* Display a message to the screen indicating how the Player object was created.

Once the loop completes, use a range-based for loop to display the contents of the array or vector, as shown in the sample output.

**General Requirements**

For complete credit, you must:

1. **MEET 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.
2. **INCLUDE 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.
3. **FOLLOW 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.
4. **SUBMIT ALL FILES BEFORE THE DUE DATE** – Submit a .zip of ONLY source code files to the dropbox for this assignment on Canvas before the due date. Do not submit anything except .cpp and/or .h, within a zip. Do not submit .exe files. Do not submit a folder structure. Do not submit project files from an IDE.

**Sample Run**

Note that this output is the result of random number generation. Your output will vary, but it should include output that explains how each Player object was created, and a neatly formatted list at the end.

A screenshot of a cell phone

Description automatically generated