CS 112 – Computer Science Fundamentals II
Assignment #5

Deadlines

This assignment is due on **Friday, December 11, 2020 @ 11:59 PM**

NOTE: No late submissions will be accepted! Solutions will be posted immediately after the due date!

How to submit your work on the assignment:

Assignments will be accepted via Canvas as CPP and H files. Use the Multiple File submission tool in Canvas to submit all files. Submit all the files in a single submission, and do NOT use ZIP.

It is expected AND REQUIRED that all the code you submit is your work and your work alone. Copying of any code between students is strictly prohibited! Also, all functions you write MUST have the correct name and headers that match the problems specifications. If the instructor's testing statements can't run your functions because of errors in the headers, that will result in major deductions in your assignment grade!

Using C++ Vectors

This assignment is designed to be an introduction to using C++ vectors in place of arrays. The focus should be in creating well-written code that meets all relevant style requirements – make this as clean and polished as you can! It will also be used in future assignments in this class, so be sure to meet this assignment's requirements as listed below.

Download the following files, adapted from the posted solutions from the Week 09 Lab:

main.cpp **PlayerChar.cpp** **PlayerChar.h** **PlayerTeam.cpp** **PlayerTeam.h**

The supplied **PlayerTeam** class definition uses arrays to hold **PlayerChar** objects. In addition, the **PlayerChar** class definition uses an array of the **PlayerType** struct to define standard player characteristics.

Refactor the supplied code to use C++ vectors instead of C++ arrays. Your refactoring must:

- Change all declaration and use of arrays to using vectors instead.
- Use vector methods such as **size()** whenever appropriate.
- No longer store the size of a **PlayerTeam** as a separate **int** value! The size of the vector is the size of the team. It still makes sense, though, to have a constructor that takes an initial team size as an **int**. By default, the code already sets all uninitialized **PlayerChar** objects to be Fred the Ogre.
- Change all functions that have arrays passed to having vectors passed instead. (remember, arrays are passed be reference by default, along with an int value containing the array's (fixed) size. When passing vectors, passing the size is no longer necessary, so that value should not be passed. Also, make sure the vectors are passed by reference!
- Allow **PlayerTeam**s to have varying numbers of **PlayerChar**s.
- Allow the programmer to add a new standard player type to the **stdPlayer** list by simply "pushing" the new standard player values onto the **stdPlayer** vector.

New methods to write include:

- **PlayerTeam::addPlayer(PlayerChar& new_player**) to add a new player to a **PlayerTeam**
- **PlayerTeam::deletePlayer(int index_to_delete)** to remove a player from a **PlayerTeam**
- While we're adding new methods, also add a method to change the **PlayerTeam**'s name
- Also add a copy constructor method to the **PlayerChar** class
- Rewrite the **==** comparison operator method for the **PlayerTeam** class to be a comparison of the total power of each **PlayerTeam** as follows:
  - The "power" of a **PlayerChar** is to be computed by multiplying its **strength** with its **agility**. For example, an Ogre starts out with **strength** 700 and **agility** 2.5, so the "power" of an Ogre with standard **strength** and **agility** is 700 x 2.5 = 1750. NOTE: use the stored **strength** and **agility** values for each individual **PlayerChar**, and not the values in the **stdPlayer** list, when calculating the power!
  - The "power" of a **PlayerTeam** is the sum of the powers of all its **PlayerChar**s.
  - If two **PlayerTeam**s have a total calculated value within 100 of each other, the **==** comparison should return **true**. If the difference in the team powers is more than 100, it should return **false**.

Update the **main.cpp** file with tests of your choice to ensure that your code is working. You do NOT need to turn in your **main.cpp** file – your code will be tested using testing code written by the instructor.