CSCI 24000 – Fall 2017
Assignment #3 – Class-y Players
Due: 10/9/2017

This third assignment will allow you to explore (by comparing and contrasting through construction and implementation) two different object-oriented programming languages (C++ and Java). You will be creating **two** separate programs – **one** written in C++ and **one** written in Java.

For this assignment, we are going to explore how we can use objects to build more expressive and cleaner programs. In honor of football season – we are going to be creating Players for our IUPUI football team (Undefeated since 1969). We will need to create and store these Players in memory and allow the user to view an entire "roster" of Players. We will make the following assumptions:

- No two Players will have the same name (*e.g., #1 Test Testerson, #2 Test Testerson*).
- No two Players can have the same number (*e.g., #1 Jane Testerson, #2 Jane Testerson*).
- Each Player MUST have at-least a **first name**, **last name**, and **jersey number**.
- In the basic submission there will only be, at maximum, **eleven (11)** Players on our team.

Your goal is to create a program that will allow for creation and viewing of Players from our "team."

Development Process:
For this assignment, all development must take place on the **master branch** in a **private** GitHub repository. You must add me and all four (4) TA's as collaborators to your repository. It is strongly recommended that you commit and push often! We will be checking to make sure that you are actively pushing changes to your repository – failure to do so will result in a deduction of points. We will also be checking that you are providing "useful" comments with your commits – failure to do so will result in a deduction of points. You are also required to include the Honor Pledge and Digital Signature in each one of your files (any file that you have written "code" in – you need to include the Honor Pledge and Digital Signature).

You are required to implement the following items as part of this assignment:

- Your C++ program must contain three files (2 .cpp and 1 .h file). These files should be named as such: **Driver.cpp**, **Player.cpp**, **Player.h**.
- Your Java program must contain the following two files: **Driver.java** and **Player.java**.
- Your Driver class must store any and all created Players in an **Array** on the **Heap**.
- This Array must make use of dynamic memory allocation.
- In the C++ version we will need to manage our memory accordingly – no memory leaks!
    - You can use Valgrind to check our program for memory leaks – any leaks in memory will result in a deduction of points.
- When you start the program, you should provide the user with a menu choice to do one of the following:
    1. Add New Player
    2. View Players
    3. Exit Program
- Upon completion of Options #1 or #2 the menu should once again be displayed to the user – thus allowing them to make another selection. This process should continue to loop until the user selections Option #3. Once/If eleven (11) Players have been entered the program should print the roster and terminate.

Below is example output of what your program should display when executed – your format must match this exactly:

```
Welcome to our CSCI 240 Roster Editor!

1) Add New Player
2) View Player(s)
3) Exit Program

Please enter your selection: 1

Please enter a first name: Ryan
Please enter a last name: Rybarczyk
Please enter a number (1-99): 15

**Player Created**

1) Add New Player
2) View Player(s)
3) Exit Program

Please enter your selection: 2

**IUPUI Football Roster**
15) Ryan Rybarczyk
************************

1) Add New Player
2) View Player(s)
3) Exit Program

Please enter your selection: 5

Invalid Choice! Please select Option #1, #2, or #3.

1) Add New Player
2) View Player(s)
3) Exit Program

Please enter your selection: 3

Thank you for using our program - Goodbye!
```

As with the previous assignments you will be submitting an algorithm. You should **always** write your algorithm before you begin actual coding. This is a good practice and demonstrates the importance of software design. If you ask for help we will always ask to see your algorithm before looking at your code. This algorithm should be the same for both Java and C++ as it should provide a generic, or abstract, view of how to solve this given problem.

Submission:

All assignments must be submitted on IU GitHub (github.iu.edu). The name of your IU GitHub repository must be as follows: **csci24000_fall2017_A3**. You should be submitting the source files for both the **C++** and **Java** versions, two **Makefiles** for the Java and C++ programs respectively (make sure to label each Makefile accordingly) that we can then use to compile and subsequently run your program, as well as a file (e.g. **algorithm.txt**) outlining your algorithm for accomplishing this assignment. Failure to follow these rules will result in a 0 on your assignment submission. Please make sure all of your files and repository are named appropriately!

Blackbelt:

For this project the Blackbelt extensions are as follows:

- Read-in data from a text file and create and store the Players in memory. (+5 Points)
- Write the "roster" out to a text file for persistence storage. (+5 Points)

****NOTES****

- You must have a working (compiling & running) version of the basic program in order for us to grade your Blackbelt submission.
- Blackbelt submissions should have their own unique algorithm, for this particular assignment, and separate source files (must be named differently than your basic submission).