# CSC 232: Data Structures and Algorithms

**Assignment 3: Convert Asn2 to Class**

**Due: See Blackboard**

**Note:** *You may make this a group project, working with ONE classmate (max group size = 2) to complete the assignment. See below for further instructions if you choose this path.*

*Submit all programs through BB by midnight on the due date. Email your programs to me* ***as a last resort*** *if you experience problems with BB. If you work in a group,* ***both members*** *should submit* ***the same*** *program.*

The purpose of this assignment is to:

* Learn how to create and use a class, its methods, and its objects
* Code a simple class, differentiating methods between the public and private interfaces
* Experiment with different designs
* Use overloading

**Requirements**

* Include the comment template at the top of your program. Use good coding practices discussed in class.
* Use only C++ language features discussed in class or presented in the book up to the date the assignment is due. Submit only what is requested.
* In Assignment 2, you wrote a non-interactive program that played a simple game called Dice Race! Start this assignment by copying Asn2.cpp to a new project and rename it Asn3.cpp.
* Your first task is to convert the game, as is, into a class called **Player** since it will represent the data and operations on each player in the game. There are many options for doing so. Experiment with different implementations to find what makes sense to you.
* Overload the << operator for printing the class objects with clean C++ syntax.
* Overload the “is equal to” operator ==. This will allow you to compare player positions with clean C++ syntax. See
  + Widely cited, thorough description of overloading operators
    - <http://stackoverflow.com/questions/4421706/operator-overloading>
  + Good chart of operators that can/cannot be overloaded
    - <http://www.tutorialspoint.com/cplusplus/cpp_overloading.htm>
* Once your code is working, add the following. Ask the user to input an odd number N. Run the program for N games, keeping track of how many times Jack won and Jill won. Store this information as a private variable in the **Player** class. Update class constructors and methods to support this new field. After all games are complete, print total games won by each player and a statement declaring one Player as the champion. It should look very close to this:

Jack won 5 games and Jill won 2 games.

Jack is the Dice Race Grand Champion!

* **If you decide to work in a group**, then discuss at least two options for how the class, methods, and objects can be structured. Implement one of the two designs and include the class declaration for it at the end of your program (commented out). Answer the following questions about the second design option (the one not implemented):
  1. What methods and objects are in this design but not the implemented design?
  2. Why did you make those choices?
  3. How did your choices affect access control to any class objects?

**Sample Output**

Roll Jack Jill

------------------------

5 5

5 2 5 Tumble Jack!

6 8

2 7

4 12

5 10 12 Tumble Jack!

2 12 11 Tumble Jill!

5 16

1 13

5 21

Jill wins game 1!