**Recitation 11**

**Topics**

* Overloading Operators

**Recitation Project**

* Design, implement and test a C++ class to work with real numbers as rational numbers.
* The data members are the numerator and denominator, stored as integers.
* Implement the operators:
  + << and >> (i.e., input and output).
    - Rational numbers are read and written as an integer, followed by a slash, followed by an integer.
    - Either (or both) of the numerator and the denominator may be input as negative integers. The following are all possible inputs: 1/2, -1/-2, -1/2 and 1/-2.
    - Only validate user input for a 0 divisor - just let your program crash if they try anything that isn't of the proper form.
    - When numbers are displayed, they must appear "normalized": the numerator and denominator will be in "lowest terms" *and* the only minus sign shown will be in the numerator. For example, if we were given as input 4/-8, we would display it as -1/2. Simililarly -8/-6 would show as 4/3. Note, we are permitted to keep the numbers normalized internally if we wish.
  + += Implement as a member function.
  + + (i.e., addition) Implement as a non-member function using the += operator. Do not make + a friend. (Note there is no reason to.)
  + ==
  + != Implement as a non-member, but not as a friend.
  + ++ and --
    - Member for ++
    - Non-member, non-friend for --
    - Both pre- and post-.
* Use separate compilation.
* Place the class in the CS1124 namespace.
* We provide a program to test your class:[testRational.cpp](http://cse.poly.edu/cs1124/Recitations/Recitation10/testRational.cpp).
* We also provide a function to compute the [greatest common divisor](http://cse.poly.edu/cs1124/Recitations/Recitation10/gcd.cpp) of two non-negative integers that should be useful for writing the normalize function. If you're clever, try to write this function yourself before looking at ours.
* You need to write the files Rational.h and Rational.cpp, satisfying the above requirements and any additional requirements indicated in the test program.
* If time allows:
  + <
  + <=, > and >= Implement as non-member, but not as a friend.