# CECS 282 - Homework 6

Complete these problems on a separate sheet of paper. Due March 10.

1. Reading from *C++ How to Program*:
   (a) Chapter 10.1, 10.3, 10.4
   (b) *Skim* Chapter 10.5, 10.6, 10.7
   (c) Chapter 10.9
2. Answer True or False for each of these questions about operator overloading. Give a one-sentence explanation for each *false* answer.
   (a) The precedence (order of operations priority) of an operator cannot be changed by overloading.
   (b) If you overload `operator==`, the compiler automatically knows how to evaluate the `!=` operator.
   (c) An operator should be a *friend operator* of a class if it does not modify the left hand side (lhs), and should be a *member operator* if it does modify the lhs.
   (d) You can modify the behavior of an operator that operates solely on primitive types, e.g., you can change the behavior of `+` when used with `int`s.
   (e) An arithmetic operator like `operator+` **must** return an object of the same type as the parameters.
   (f) You *should* overload every operator for every class you write.
3. C++ provides built-in arithmetic operators for the various primitive types. Suppose you have declared `int x`, `double y`, and `char z`. For each of the following examples, write the function header for an operator that would work in the statement given. One answer is supplied:
   (a) `x + y`
   Ans: `double operator+(int lhs, double rhs)`, since the lhs is an int, the rhs is a double, and by the rules of the language, the result will be the more-precise data type (double).
   (b) `-x`
   (c) `y / x`
   (d) `x >> z`
   (you may need to look up the meaning of `operator >>`; it is not solely for `cin`)
   (e) `z * y`
4. Explain the difference between the post-increment and pre-increment operators in C++. Write a function declaration/header for both operators with the `Rational` class from Lab 5.
5. Read about the C++11 Standard Library class `std::array`. Answer the following questions:
   (a) Show how to declare a `std::array` with 10 elements, containing the numbers 1 through 10.
   (b) Show how to change the element at index 3 to `100`.
   (c) Show how to declare an 8x8 **2-dimensional array** using the `std::array` type.
   (d) **Without using the `[]` operator**, show how to access row 3, column 2 in your 2-dimensional array.
   (e) True or false: the size of a `std::array` can be chosen at **runtime**.
   (f) **Why is there a `std::array` class in the first place, if C-style arrays seem to work just fine?**