CS 3210, Lab #2, Class Design and Implementation

# Purpose

The purpose of this lab will be to explore overloaded operators to create a first-class type in C++.

# Assignment

Download the "lab2starter.zip" file in the shared Box folder. Unzip and verify that you can compile the skeleton code. We will discuss implementation of the constructors, destructor, assignment operator, and perhaps a few miscellaneous other methods in lab. You will be responsible for completing the implementation and fully testing it. This code will be used in future lab assignments, so it must be 100% functional.

This assignment is due to be submitted prior to lab in Week 3.

# Hints

* Do not change any of the public methods in the header file unless instructed to do so.
* You may add private “helper” methods as needed.
* Be careful with the return types - note that the operators return objects, not pointers or references. Is this efficient?
* Note the const items.
* Do not change any of the public methods in the header file unless instructed to do so, or to correct "deficiencies" noted in the comments.
* Note that some methods are commented that they should throw exceptions. Why these? What other options are there for reporting errors?

# Submission

Prior to lab in Week 3, submit a zip file with your complete code. You may be asked to demonstrate your program to the instructor during lab.

# Grading Checklist

* Submission received on time (grade multiplier 0.80 if not)
* Check items that will apply to all assignments (50 pts, 10 pts each)
  + Submission complies with guidelines (color pdf, no wrapped lines, line numbers, proper ordering, zip file with buildable code)
  + Source code presents as professionally written (comments, indentation, variable naming, structured)
  + Compiles with no errors or warnings (using -Wall)
  + Runs to completion with no crashes or memory leaks (using valgrind)
  + Makes exclusive use of C++ standard library features - no libc unless necessary
* Check items that are particular to this assignment (50 pts, divided equally unless otherwise noted)
  + No changes are made to public interface of matrix class
  + All methods are implemented correctly (higher weight)
  + All methods are reasonably tested by driver code in main and/or functions called from main
  + Exceptions are thrown when error conditions exist per comments - this is tested as well
  + Execution of tests are reasonably verbose