CFRM 520 Syllabus

Financial Software Development and Integration with C++

Instructor: Daniel Hanson ([hansondj@uw.edu](mailto:hansondj@uw.edu)), Lewis Hall 230F
Office Hours: Tues/Thurs 11.30 am - 12.30 pm, and by appointment/online

Textbook: None specifically required; three recommended texts, however, are as follows, and will referenced with some frequency throughout the course:

1) S Rao: *Sams Teach Yourself C++, 7th Edition* (SAMS). This is a pretty good overall introductory text on C++, which includes topics on the C++11 Standard.

2) Nicolai Josuttis: *The C++ Standard Library, 2nd Edition* (Addison Wesley). This is really the best book of all, in my opinion, on the C++ Standard Library. I used the 1st edition extensively in practice, and now the newer edition includes all the features of C++11. It is not imperative that you have it for the class, but if you intend to pursue C++ as part of your career, this book is a must-have.


Additional books and other sources will be referenced in the lectures and will be announced as we move forward. Unfortunately, there is no one single good book available for quantitative financial development in C++. When available, web links will be provided.

Video recordings of lectures are made available after each lecture and will be available under the Panopto Recordings section on Canvas.

Required computer resources: A portable computer with a C++ compiler that (at least) includes the recent C++14 standard. Visual Studio 2017 is highly recommended, particularly for an introductory course, due to its development and debugging tools. It is what I will mainly be using in presentations; furthermore, toward the end of the quarter, we may look at topics such as integration with Excel and Microsoft .NET (both are commonly used in quant finance), in which case the Visual Studio compiler and tools will be mandatory. A working R environment, with the rtools package, will also be required when we discuss the Rcpp package.

Course Topics:
The course will be broken up into four parts: 1) **C++ Basics**, 2) **Numerical C++ and Foundations for Quant Finance**, 3) **Financial Maths in C++**, and 4) **Useful Utilities for C++ Quant Development/Advanced Topics in C++**.

1) **C++ Basics**
   a) Introduction (Jon Kalb lecture and pdf)
   b) Procedural programming in C++
   c) Compiling programs
   d) Conditional and iterative statements (if conditions, for loops)
   e) Functions
   f) Object Based, and Object Oriented Programming in C++
   g) Pointers, smart pointers, and memory management
   h) STL containers (eg vector), iterators, and algorithms (very important topic)
   i) Function objects, lambdas, and using them with STL algorithms
   j) Move semantics

2) **Numerical C++ and Foundations for Quant Finance**
   a) Random number generation
   b) Multithreading with the new C++11 Standard future class
   c) Matrix libraries (Eigen)
   d) Dates and daycount rules
3) **Financial Maths in C++**
   a) Implementing term structures
   b) Closed form pricing models: bonds and options
   c) Monte Carlo based option pricing and risk management
   d) Empirical financial models -- hedge fund style analysis (time permitting)

4) **Useful Utilities for C++ Quant Development/Advanced Topics in C++**
   a) Rcpp: an R package for calling C++ from R scripts
   b) Introduction to design patterns
   c) Templates and static polymorphism
   d) Best practices (will be reinforced throughout the course)
   e) XLW: a utility that provides an interface between Excel and C++ (tentative/time permitting)
   f) Integration with C# and .NET (tentative/time permitting)

There will be a midterm exam that will mainly cover Part 1 of the course, and possibly a comprehensive in-class final exam at the end of the course, or a final project or higher-weighted assignment.
Weekly assignments and quizzes will also be given. The assignments are especially important, as this is the best way to truly learn how to program in C++.

**Assignment Grading:** Students are explicitly advised that their software source code may be subjected to a Measure of Software Similarly checker that compares every submission of every student against every other submission and alerts the instructor to potential software plagiarism. Plagiarism is a serious violation of basic academic standards. Students found to be involved in software plagiarism (recipients or suppliers) will receive zero credit and may also face other academic misconduct discipline; see academic integrity section below.

**Policy on late assignments:** Assignment due dates are hard deadlines and no late assignments will be accepted. Extenuating circumstances (documented medical issue or death in the family) will be handled on an individual basis.

**Grading:** Grades will be based on a combination of scores on homework assignments, weekly quizzes, a midterm exam, and perhaps a final project or in-class final exam, plus attendance and class participation. Weighting of each component is as follows:

- Midterm exam  30%
- Final Exam or project 35% (alternatively, a higher-weighted homework assignment may be given)
- Assignments 35% (mostly weighted toward assignments)

Class Participation and Attendance: Class participation is encouraged and may affect borderline grading decisions; this includes participation on the Canvas Discussions forum, and the participation rate in course evaluations.

*** NOTE: Attendance is required for resident students ***