NYU FRE Department
FRE-GY 6883 Financial Computing

Song Tang, st290@nyu.edu, 646-283-4578

Overview:

This course covers programming applications to financial engineering, using C++ in Amazon AWS as the development environments for them. Topics include structured and object-oriented programming in C++ with applications to binomial options pricing, calculation of implied volatility, option pricing via Monte Carlo methods, market data access libraries with applications to historical financial data series retrieval and management, and other advanced programming concepts important for financial engineering such as numerical techniques, trading systems, and large-scale software design.

Schedule of Classes:

Week	Topics
Part 1: Using C/C++ for Financial Computing	
1	Topic 1: Set up AWS Linux Virtual Machine C++ IDE Environment
2	Topic 2: Structured Programming – Binomial Tree Model Implementation
3	Topic 3: Function Pointers – CRR Pricer for European Call and Put
4	Topic 4: Object-Oriented Programming – Binomial Tree Mode Class
5	Topic 5: Inheritance and Polymorphism – European Option Pricing Framework
6	Topic 6: Data Structures– Pricing American Options
7	Topic 7: Class Templates – Price Tree and Stopping Tree for American Options
8	Topic 8: Implementing Non-linear Solvers using Function Pointers and Inheritance
9	Topic 9: Monte Carlo Methods for Path-dependent Options
10	Topic 10: Enhancing Monte Carlo Framework for Pricing Basket Options
Part 2: Develop Financial Applications in C/C++	
11	Topic 11: Integrating Historical and Market Data in Financial Application
12	Topic 12: Trading Systems & Large-Scale Financial System Design
13	Review for Final Exam
14	Final Exam
15	Team Project Presentation and Demonstration

Class Meeting Times:

- Instruction Mode: In Person
  - Tuesdays, 6:00pm – 8:30pm, Rogers Hall Rm 304
  - Saturdays, 1:30pm – 4:00pm, Rogers Hall Rm 302
- Office Hours: after each session and by appointment

Assessment:

Students will do extensive coding for both their homework assignments and the final project. Homework assignments are requested to be completed independently. Homework assignments are closely related to the topics and programs discussed in classes. The final project will be done in groups of 5 students. The final project will be given to students after part 1 is completed, so students will have 5 weeks working on their projects. The last class will be the project presentation and program demonstration.

The assessment will be done as the following:

- Quizzes, 10%
- Homework Assignments, 30%
- Final Exam, 30%
- Group Project, 30%

Class Materials and Textbooks:

(1) Lecture Slides and Source Codes. It is highly recommended to take lecture notes as we will add a lot of information and coding details through lectures.
(2) Numerical Methods in Finance with C++ (Mastering Mathematical Finance), by Maciej J. Capinski and Tomasz Zastawniak, Cambridge University Press, 2012, ISBN-10: 0521177162
(3) Introduction to C++ for Financial Engineers: An Object-Oriented Approach (The Wiley Finance Series), by Daniel J. Duffy, Wiley, 2006, ISBN-10: 0470015381

Development Environment:

- Use GCC in Amazon AWS Linux virtual machine development environment.

Course Topics:

Set up AWS Linux Virtual Machine C++ IDE Environment (Week 1)

- General introduction of C++
- Introduction of AWS Cloud 9
- AWS Account Creation
- Setup GitHub Account
- Join FRE6883 GitHub Classroom

Pricing European Options in C++ (Week 2-5)

- The Structured Programming
  Structured programming is a technique which arose from the analysis of the flow control structures which underlie all computer programs. It is possible to construct any flow control structure from three basic structures: sequential, conditional, and iterative. Students will learn how to price European options via the binomial model.
  Topics covered in this section:
  - Principle of Structured Programming
  - Flow Control Structures
  - Arrays
  - Functions and Function Calls
  - Separate Compilation
  - Cox-Ross-Rubinstein (CRR) Pricer
  - Pointers and Function Pointers.

- Object-Oriented Programming in C++
  In this unit students will gain a basic understanding of object-oriented programming using C++ and the design of a C++ application based on what they did in structured programming. In other words, they will recast their option pricer in the style of object-oriented programming. Their classes will reflect the relationships between real entities, namely the binomial model and European options of various kinds.
  Topics covered in this section:
    - Our First Class
    - Inheritance
    - Virtual Functions
    - Recast the Option Pricer

Using Data Structures and Class Templates for American Options (Week 6-7)

This unit will introduce students advanced object-oriented programming techniques and demonstrate how these techniques could be applied to solve complicated problems in quantitative finance. Students will learn how to implement American options by using C++ data structures and class templates.

Topics covered in this unit:

- C++ STL Data Structures
- Template Functions and Class Templates
- Computing Option Price via Black-Scholes Formula

Implementation of Non-linear Solvers in C++ (Week 8)

This unit will use C++ function pointers, virtual functions, function templates, and design patterns to implement nonlinear solvers.

Topics covered in this unit:

- Bisection Method
- Newton-Raphson Method
- Function Pointers
- Virtual Functions
- Function Templates
  Computing Implied Volatility

Monte Carlo Methods (Week 9-10)

This unit covers basics of Monte Carlo simulations for path-dependent options, with emphasis on practical issues such error analysis, variance reduction and algorithm updates.

Topics covered in this unit:

- Path-dependent Options
- Valuation
- Pricing Error
- Greek Parameters
- Variance Reduction
- Path-dependent Basket Options

Integrating Historical and Market Data in Financial Applications (Week 11)

This unit will let students gain an understanding of financial market data processing. It will teach student how to build visualization in a C++ application by integration with gnuplot. In addition, it will demonstrate to students how to fetch historical market data using libcurl in C++.

Topics covered in this unit:

- Access Market Data using libcurl in C++.
  - Handle the Easy libcurl.
  - Fetch historical stock data for market data sources.
- Integrate gnuplot into a C++ application.
  - Setup AWS Linux Virtual Machine for gunplot
  - Use gnuplot in C++ program for result visualization.

Trading Systems & Large-Scale Financial System Design (Week 12)

This unit is to introduce students to the concepts, terminology and code structures required to develop trading applications, as well as high frequency trading and dark pool. In addition, this unit will show students a comprehensive top-down approach to the logical design of individual components for large-scale financial projects.

Topics covered in this unit:

- High Frequency Trading
  - US Equity Share Volume by Market Participant
  - US Equity Trading by Market Participant
  - Challenges in High-Frequency Trading
  - HFT Technologies
  - Dark Pools
- Software Development Cycle
  - Ad-hoc Development
  - Lifecycle Stages
  - Benefits and Limitations of Lifecycle Models
  - Code-and-fix Model
  - Waterfall Model
  - Spiral model – Risk Oriented
  - Staged Delivery Model
  - Evolutionary Prototyping Model
  - Design-to-Schedule
- Trade System Development Cycle
  - Key Ideas in K/V Software Methodology
  - Advantages of K/V Development
  - K/V Trading System Development Stages

Course Team Projects:

Students are required to do class projects in groups. Groups once formed cannot be changed midway through the project. The team lead is responsible for facilitating the planning of the project, and the entire team will plan the project under the guidance of your team leader. Planning involves identifying what should be done (tasks), who should do it (resources), when tasks should be done (time frames) and how tasks are best sequenced (dependencies).

Each team will submit project report and source codes tar/zipped via email before the deadline. The project reports should include a brief executive summary, the project design approach, design choices and implementation specifics. All the teams are requested to present and demonstrate their projects. The details of the team project will be announced and posted on our course site once part 1 is completed.

Letter Grades:

Letter grades for the entire course will be assigned as follows:

Letter Grade	Points	Percent
A	4.00	93.33%
A-	3.67	90.00%
B+	3.33	86.67%
B	3.00	83.33%
B-	2.67	80.00%
C+	2.33	76.67%
C	2.00	70.00%
F	0.00	0.00%

Policies:

Academic Misconduct

A. Introduction: The School of Engineering encourages academic excellence in an environment that promotes honesty, integrity, and fairness, and students at the School of Engineering are expected to exhibit those qualities in their academic work. It is through the process of submitting their own work and receiving honest feedback on that work that students may progress academically. Any act of academic dishonesty is seen as an attack upon the school and will not be tolerated. Furthermore, those who breach the school's rules on academic integrity will be sanctioned under this Policy. Students are responsible for familiarizing themselves with the School's Policy on Academic Misconduct.
B. Definition: Academic dishonesty may include misrepresentation, deception, dishonesty, or any act of falsification committed by a student to influence a grade or other academic evaluation. Academic dishonesty also includes intentionally damaging the academic work

of others or assisting other students in acts of dishonesty. Common examples of academically dishonest behavior include, but are not limited to, the following:

1. Cheating: intentionally using or attempting to use unauthorized notes, books, electronic media, or electronic communications in an exam; talking with fellow students or looking at another person's work during an exam; submitting work prepared in advance for an in-class examination; having someone take an exam for you or taking an exam for someone else; violating other rules governing the administration of examinations.
2. Fabrication: including but not limited to, falsifying experimental data and/or citations.
3. Plagiarism: Intentionally or knowingly representing the words or ideas of another as one's own in any academic exercise; failure to attribute direct quotations, paraphrases, or borrowed facts or information.
4. Unauthorized collaboration: working together on work that was meant to be done individually.
5. Duplicating work: presenting for grading the same work for more than one project or in more than one class, unless express and prior permission have been received from the course instructor(s) or research adviser involved.
6. Forgery: altering any academic document, including, but not limited to, academic records, admissions materials, or medical excuses.

Disability Disclosure Statement

Academic accommodation is available for students with disabilities. Please contact the **Moses Center for Students with Disabilities** (212-998-4980 or mosescsd@nyu.edu) for further information. Students who are requesting academic accommodations are advised to reach out to the Moses Center as early as possible in the semester for assistance.

Inclusion Statement

The NYU Tandon School values an inclusive and equitable environment for all our students. I hope to foster a sense of community in this class and consider it a place where individuals of all backgrounds, beliefs, ethnicities, national origins, gender identities, sexual orientations, religious and political affiliations, and abilities will be treated with respect. It is my intent that all students' learning needs be addressed both in and out of class, and that the diversity that students bring to this class be viewed as a resource, strength and benefit. If this standard is not being upheld, please feel free to speak with me.