I. Administrative Details

- **instructor:** Aaron Stevens, azs@bu.edu; Office: QST 546F
- **office hours:** Wednesday 2-3pm and Friday 1-2pm
- **pre-requisites:** This class is suited for an advanced undergraduate student in the Finance concentration, or a student majoring in computer science with a strong interest in Finance. Prerequisite courses will include the Core class in Financial Management (FE323) and an introductory computer-programming course (CS108 or CS111 or equivalent). Students are recommended to take the Investments class (FE445) before taking FE459, or else be prepared to do independent readings before the class begins.

- **Course schedule:** Wednesdays and Fridays, 9:30am – 1:00 pm

II. Course Description, Learning Goals, Structure, & Materials

II.1. Course Description & Learning Goals

The course will teach students how to use computational techniques to implement financial algorithms for security pricing and risk analysis, including, bonds, stocks, and options. This will be a rigorous, hands-on programming course to prepare students for quantitative jobs in finance. The overall objective of the course is to enhance the students’ understanding of the well-known financial models used to price securities including bonds and options, to evaluate the risk and return characteristics of stocks and portfolios. After the course, students will have a deeper understanding of investment portfolios, risk management techniques that use derivatives, and arbitrage strategies. Additionally, students will become comfortable with a modern programming language based on functional and object-oriented programming, which will enhance their job opportunities in a variety of fields beyond finance.

Course Learning Objectives

- Use computer programming to implement well-known financial algorithms
- Apply computational modeling to analyze investment risk and return
- Develop data analysis and back-testing skills
- Develop programming experience in functional and object-oriented programming

II.2. Course Structure & Pedagogy

Learning to program is a skill that takes practice, similar to learning a sport or a musical instrument. You cannot learn to play basketball by simply watching the Celtics; rather, you need to go to the gym and practice your shots. This course is designed around a set of weekly assignments, which you must
complete in order to master the material. It is not possible to learn how to program by simply “watching.” To become a programmer, you must actively program!

The course will be a rigorous and **hands-on programming experience**. The class meetings will be designed around a series of practical assignments. Each assignment will build a concrete implementation of an element of finance theory and develop a computational approach to problem solving. The course will use a **flipped-classroom methodology**, in which traditional uses of “lecture” and “homework” have been exchanged. Students will prepare for each class with a review of programming and financial concepts via readings and videos of mini-lectures and programming examples. In-class time will be used primarily for individual and small-group programming exercises and assignments.

Assignments will often be very time consuming, so you must plan ahead and start early. Each week’s assignment is due on the Sunday following the week of class and is a necessary building block for the following week’s material.

### II.3. Books and Readings:

- *What Hedge Funds Really Do*, by Philip Romero and Tucker Balch. (denoted *Hedge Funds*)
  

- *Essentials of Investments*, by Zvi Bodie, Alex Kane, and Alan Marcus, 9th edition, coordinated with FE445. (denoted *Investments*)
  

- *How to Think Like a Computer Scientist with Python 3* by Peter Wentworth, Jeffrey Elkner, Allen B. Downey, and Chris Meyers. (denoted *Python 3*)
  
  Free online book: [http://openbookproject.net/thinkcs/python/english3e/](http://openbookproject.net/thinkcs/python/english3e/)

### II.4. Software:

We will use the *Spyder IDE* (Integrated Development Environment) for Python. Download/install from the Anaconda scientific computing package: [https://www.anaconda.com/](https://www.anaconda.com/).

### II.5. Learning Management System:

All materials will be posted or linked to Questrom Tools, [http://questromtools.bu.edu](http://questromtools.bu.edu).
III. Course Expectations and Policies

III.1. Attendance Policy

- Satisfactory class contributions require attendance at every session of the course; preparation of all materials for every session; and active, quality participation in class discussions. Simply attending class, however, does not constitute a positive contribution to class and will not yield high class contribution scores. Recognizing that you are facing complex demands on your time, we can excuse two absences during the term (for any reason). If you anticipate that you will need to miss three sessions or more, then you should take this course in another term. Students who miss 3 or 4 sessions will lose 3 and 6 points on their final course grade, respectively. Students who miss more than 5 sessions will have their final grades reduced an entire letter grade and students who miss more than 7 sessions may fail the course as a result. Assignments are always due at the beginning of class on their due date, even if students are unable to attend class that day.

III.2. Academic accommodations for students with special needs:

- In keeping with University policy, any student with a disability who needs or thinks they need academic accommodations must call the Office of Disability Services at 617-353-3658 or stop by 19 Deerfield Street to arrange a confidential appointment with a Disability Services staff member. Accommodation letters must be delivered to your instructor in a timely fashion (not later than two weeks before any major examination). Please note that accommodations will not be delivered absent an official letter of accommodation.

III.3. Collaboration and Academic Integrity Policy

- Students are encouraged to discuss assignment requirements, formulas, sample outputs, and algorithms as part of the learning process. Students are always encouraged to discuss specific questions about programming syntax and error messages. However, all programming assignments are expected to be individual work.

- With respect to FE459, plagiarism is specifically defined to include (but is not limited to) the following:
  - collaboration on the code you write (unless you are specifically instructed to do so on a specific task)
  - copying any part of someone else's program, even if you have permission and/or have modified the code
  - sharing or giving your code or even a subset of your code to another student to review
  - reviewing another student’s solution (including from past semesters)

It is our course policy to use automatic plagiarism detection software. Suspicious similarities will be uncovered. It is the student’s responsibility to know and understand the provisions of the Questrom School of Management’s Academic Conduct Code, copies of which are available here: http://questromworld.bu.edu/acc/
IV. Course Evaluation & Expectations

IV.1. Course Evaluation

Grading: The relative weighting of assignments in your course grade is as follows:

<table>
<thead>
<tr>
<th>Evaluation Activity</th>
<th>Fraction of Course Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regular Weekly Assignments</td>
<td>40%</td>
</tr>
<tr>
<td>Final Project</td>
<td>10%</td>
</tr>
<tr>
<td>Midterm Exam</td>
<td>20%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>20%</td>
</tr>
<tr>
<td>Participation, Attendance, and Class Contributions</td>
<td>10%</td>
</tr>
<tr>
<td>TOTAL</td>
<td>100%</td>
</tr>
</tbody>
</table>

To pass the course, you must earn a passing grade for each of the five components.

There will be absolutely no grade deflation in this course. (There will also, however, not be any grade inflation.) Final course grades will be the result of a direct mathematical computation, based on the formula above.

If you have any questions about grades that you receive on particular assignments, you must raise them within two weeks of receiving your grade on that assignment. Unless we made computational errors, we will be unable to alter grades after final grades have been determined. If you have particular grade-related considerations that you think are important, please raise these with your instructor as early as possible (during the first half of the semester at the latest!), so that your instructor can help you approach the course in a way that will help you achieve your best possible performance.

IV.2. Assignment Descriptions & Expectations

- Please provide at least a cursory description of each assignment/grade component, along with any relevant notes or explanations.

- If the general Academic Integrity statement does not clarify expectations for each assignment, please state your academic integrity expectations for each assignment here: For example, can students work together on the assignments or should they work individually? If assignments are Individual Assignments, can students talk about them together? Can teams talk with other teams about their progress?

- Please note for each exam:
  - what is the general format (e.g., multiple choice, essay, etc.; open-book, closed-book)?
  - what %, if any, will be multiple choice?
  - will it be cumulative?
  - will it be returned to students for their retention?
  - will old versions be posted for review?

- If a project is part of the course, will it be a Team or Individual Project:
  - if there is a Team Project, how will teams be assigned (i.e., randomly, self-selected, faculty-constructed, balanced by concentrations, etc.)
  - note that the UGPDC strongly recommends that teams not be constructed using a public “Four Corners Approach” (i.e., an approach that divides students based on observable characteristics, such as US-based vs. International Students)

- If class contributions or class participation constitutes 15% of the course grade or more…
  - please describe how points are assigned (and how lateness/absences affect this score)
  - please plan to provide mid-semester feedback (e.g., post scores and/or meetings)
V. Course Schedule
The course is broken down in 4 logical modules briefly motivated below their title.

Module 1: Warm Up: Programming for Investments Valuation
Implementing the power of Python without using its most complicated features for standard finance and data analysis applications.

Session 1: INTRODUCTION (Wednesday 5/23/18)
PYTHON REVIEW AND BASIC FINANCE CALCULATIONS
Assignment 1: Time Value of Money Functions and Iterative IRR Calculations
Reading #1: Hedge Funds, Chapters 1-4
Reading #2: Investments, Chapters 1 and 2
Reading #3: Python 3, Chapter 1-4 (Python Review)

Learning Goals and/or Class Preparation Questions:
1. What is programming for finance?
2. What are hedge funds, and who works there?
3. What is a quant, and what does a quant do?
4. Finance Review: Present Value, Future Value, Rates of Return (simple, IRR)
5. Introduction to Spyder: our programming environment
6. Python review: Basic data and functions
7. Deeper Python: Iterative Programming

Session 2: HISTORICAL RATES OF RETURN (Friday 5/25/18)
DESCRIPTIVE STATISTICS FOR STOCKS
Assignment 2: Descriptive Statistics for Stock Market Data
Reading #1: Investments, Chapters 5
Reading #2: Python 3, Chapter 7.1-7.3 Iteration (for loop), 7.8, 7.10
Reading #3: Python 3, Chapter 11 Lists, Sections 11.10-11.13
Reading #4: Python 3, Chapter 13 (Files), Sections 13.1-13.4

Learning Goals and/or Class Preparation Questions:
1. Python list operations and iteration
2. The accumulator design pattern
3. Calculating financial returns from historical prices
4. Obtaining historical stock price information
5. Processing CSV data files in Python
7. Estimating parameters for CAPM: beta and Jensen’s alpha

Session 3: BOND PRICING AND YIELD CALCULATIONS (Wednesday 5/30/18)
BOND RISK AND RETURN
Assignment 3: Bond Pricing, Yield, and Risk Metrics
Reading #1: Investments, Chapter 14
Reading #2: Python 3, Chapter 7.1-7.3 Iteration (for loop), 7.8, 7.10

Learning Goals and/or Class Preparation Questions:
1. Finance: Relationship between price, coupon rate and yield
2. Develop functions for bond price, yield, calculations
3. Implementing a complete bond price calculator
4. Sensitivity of bond prices to changes in interest rate (maturity, interest rate)
5. Metrics to quantify bonds’ interest rate risk: duration and convexity
6. Developing a full report illustrating a bond’s risk
NO CLASS ON FRIDAY 6/1/18; SUBSTITUTE MONDAY SCHEDULE ON FRIDAY 6/1/18

Module 2 Strength Training for Programmers: Matrix Operations and Object Orientation

Building needed fundamentals of linear algebra and matrix operations in Python, as well as object orientation, before applying these methods to finance problems. Computations in Modules 3 and 4 (optimal portfolios, option pricing) would not be practically feasible without the skills of Module 2.

Session 4: 2-DIMENSION LIST DATA (Wednesday 6/6/18)
MATRIX OPERATIONS
Assignment 4: Matrix Operations; Bond Pricing
Reading: Python 3, Chapter 11 Lists (2 dimensional)

Learning Goals and/or Class Preparation Questions:
1. Modeling 2-dimensional data series as Python lists
2. Review of matrix algebra operations
3. Implementing matrix operations in python functions
4. Application: bond pricing via matrix multiplication

Session 5: OBJECT-ORIENTATION (Friday 6/8/18)
Assignment 5: Object-Orientation
Reading #1: Python 3, 15. Classes and Objects (all)
Reading #2: Python 3, 21, More object oriented programming

Learning Goals and/or Class Preparation Questions:
1. Object-oriented programming: thinking in objects
2. Modeling object data and methods
3. Creating instances of objects
4. Object methods: operator overloading
5. Developing a data panel object for financial computing
6. Application: Financial returns and descriptive statistics using OOP

Module 3: Using Your Strength: Options Pricing Algorithms

Being able to understand and implement the famous (at least in Finance!) pricing trees is a sought after skill in the financial industry.

Session 6: BINOMIAL STOCK PRICE TREES (Wednesday 6/13/18)
BINOMIAL OPTIONS PRICING
Assignment 6: Binomial Stock Trees and Binomial Option Pricing
Reading #1: Python P3, Chapter 11 Lists (2 dimensional)
Reading #2: Investments, Chapter 20

Learning Goals and/or Class Preparation Questions:
1. Overview of risk neutral probabilities
2. Binomial trees: stock pricing
3. Programming with binomial trees
4. Key characteristics of European/American call/put options?
5. Payoff functions for classic calls / puts and option strategies.
6. How does the binomial option pricing method approximate the option prices?
Session 7: BLACK-SCHOLES OPTIONS PRICING *(Friday 6/15/18)*
IMPLIED VOLATILITY

Assignment 7: Option Pricing with Black Scholes
Reading #1: Investments, Chapter 21
Reading #2: Python 3, Chapter 23 Inheritance (all)
Reading #3: on the VIX, the fear index (TBD)

Learning Goals and/or Class Preparation Questions:
1. Implementing the Black-Scholes option pricing formula
2. Creating an object-oriented model for option pricing
3. Demonstrating how parameters affect the option price: the *Greeks*
4. Time varying volatility in the stock market
5. Calculating implied volatility from market option prices
6. Evaluating arbitrage opportunities in option prices

Session 8: MONTE CARLO SIMULATION *(Wednesday 6/20/18)*
EXOTIC OPTIONS

Assignment 8: Monte Carlo Simulation
Reading #1: Investments, Chapter 21
Reading #2: Numpy Documentation (mathematical functions)
Reading #3: PyPlot Documentation (advanced plotting functions)
Reading #4: Wikipedia: Exotic Options

Learning Goals and/or Class Preparation Questions:
1. Introduction to Numpy data objects
2. Generating simulated stock returns
3. Graphing simulated stock returns
4. Using Monte Carlo simulation to price standard options
5. Describing exotic options by their payoff rules
6. Implementing exotic options algorithms

Module 4: The Big Game: Portfolio Construction and Hedging
If you are a finance quant and are not busy pricing an option, you are probably constructing an optimized portfolio, evaluating portfolio managers’ performance. It is the point of Module 4!

Session 9: NUMPY AND PANDAS *(Friday 6/22/18)*
THE EFFICIENT FRONTIER

Assignment 9: Creating Efficient Portfolios
Reading #1: Numpy documentation
Reading #2: Pandas documentation
Reading #3: Investments, Chapter 7

Learning Goals and/or Class Preparation Questions:
1. Linear algebra with Numpy
2. Gathering financial data with Pandas
3. Graphing historical financial data
4. Defining portfolio efficiency in Mean vs variance
5. Calculating minimum variance and maximum Sharpe ratio portfolios
6. Graphing the efficient frontier
CHOOSE AMONG THESE REMAINING TOPICS FOR SESSIONS 11 AND 12

Session 11: MARKET EFFICIENCY (Wednesday 6/27/18)
BACK TESTING
Assignment 11: Event Studies and Back Testing
Reading #1: Hedge Funds, Chapters 3-8, 9, 11, 14
Reading #2: Investments, skim Chapter 11, skip Chapter 12

Learning Goals and/or Class Preparation Questions:
1. How should (do??) markets respond to new information?
2. Do asset prices fully and quickly incorporate news?
3. Implementing an event study, abnormal results and market efficiency
4. What is back testing?
5. Defining a strategy to test the profitability of a trading rule
6. Simulating a trading rule
7. Visualizing your results and performance

Session 12: FINAL EXAM IN CLASS