1. Goal: Design a powerful quantitative finance library (qfl) using C++ which contains most popular derivative products and pricing models. The library should be well-designed, comprehensive, robust, and easy to use.

2. Grading rule: report (10%), minimum level accomplished (60%), one of the extension level (A, or B) accomplished (30%), something very fancy beyond above (Bonus point plus 5% - 30%).

3. Report: A “readme” document or report is required (amount to 10% of total points) which must at least contains the following:
   • A clear description of the logic of the design of the library
   • All the financial products (and their properties) or models the library supports
   • A detailed list of all objects (classes, functions, templates, etc.) which shows the syntax of every object, the arguments of the object, the meaning of each parameter or member function, and possible issues (limitations) while using the object.
   • A sample ”main” function which calls most of the objects in the library and prints out important results for validation purpose.

4. Minimum level: amount to 50% of total points. You are required to at least include the following objects or functions in your library:
   • Underlying asset: stock with no dividend
   • European type put and call option: necessary parameters, BS prices, MC prices, PDF prices(using finite different method), greeks (at least Delta)
   • American type put and call option: necessary parameters, MC prices, PDF prices
   • discrete-time models (such as binomial tree) for European type options and American type options
   • implied volatility calculations

5. Extension level A: Extend the library to include interest rate models. To get full points of this extension, you should at least cover the following:
   • short rate models (allowing a general affine type term structure)
   • forward rate models (HJM framework)
   • Libor market models with simple volatility function shapes
   • zero coupon bonds, coupon bonds, caplet, swaps (prices, term structures, durations of those products)
   • recover the yield curve or term structures given necessary prices as input
• you can make some assumptions or simplifications as you need, but it must be expressed clearly and reasonable.

6. Extension level B: Design some kind of “patterns” or operators” which can calculate prices for any given general payoff. The only thing needs to be specified is the type of the derivative (American, European, Asian or some other path dependent types). The goal is to offer as much flexibility as you can while the basic types of equity derivatives such as call and put are still included. The library should then be much easier to use and you should successfully handle more functioning with fewer objects or functions. You should not treat the minimum level and this level as two separate work. Instead, I expect you incorporate or unite every thing together into one consistent library. Possible techniques you might need to use are abstract classes, template classes/functions, point functions, etc. Possible routines you may try: design of ”risk-neutral expectation” operator, design of template pricing functions/algorithms, design of abstract/template class containers which can contain all kinds of products, design of abstract binomial tree patterns, etc.

7. If you decide to do a project with a different topic, please let me know ASAP. You should not take the risk to do a totally different project without telling me in advance.