# 46-935 Homework Assignment #2

Short Answer (40 points total):

1. Recall that in the last assignment we attempted to use a binary tree to improve the efficiency of the linked-list insertion operation. Assuming a reasonably balanced tree, the BSTList unique insertion method should have taken on average O(log n) comparisons. However, in practice we find that the binary search trees generated by our BSTList insertion algorithm are NOT reasonably balanced. Why aren't they balanced? What effect does this have on the number of comparisons needed for unique insertion? Describe modifications to our insertion algorithm that you think would help fix this problem.

2. What is the difference between a FileInputStream, InputStreamReader, and a BufferedReader? What is a string tokenizer (java.util.StringTokenizer)?

3. More Java I/O

   File numbers.txt:

   ```
   10  0.100  3.000
   ```

   ```
   Class Holder
   {
       public int time;
       public double short_rate;
       public double option_value;
   }
   ```

   Write java pseudo-code to fill in the members of the Holder class with data read in from the numbers.txt file. The first number in the file is the time, followed by the short rate and option value.

   ```
   public void readAndParseFile(Holder objToFill)
   {
       //your code goes here
   }
   ```

4. Imagine that you wished to use the optionCalc package (from the last assignment) to price options whose payoff depended on other options as auxiliary processes. If this was the case, in the AbstractOption class we would need to keep a Vector (java.util.Vector) of references to other AbstractOption objects (the dependencies).

Each of the dependencies may in turn have AbstractOption objects upon which it is dependent. This sequence of dependencies can be viewed as a graph. What needs to be true of the dependency graph of an option to allow calculation of its payoff? What modifications would you need to make to the AbstractOption class to be able to compute the payoff of these types of options?

Programming (60 points total):

Download the source files from the Assign2 handout directory via FTP (/afs/andrew/course/46/935/handout/Assign2) or download the zip file from the 46-935 web page. You will not use all of the files.

I've included all of the source files that we have been discussing in class for your reference. In this assignment, we will only be using the files in the assign2 and mathUtil packages.

You must implement the BlackScholesCallObject and BlackScholesPutObject classes in the assign2 package (skeletons have been provided). These classes extend the AbstractFunctionObject class to implement the Black-Scholes formula for call and put options. You should add member fields to these classes to hold the values that you will need in the evaluate method (strike price, share price, time to expiration, interest rate, observed option price).

The evaluate method takes the volatility as a parameter (val[0]) and calculates the difference between the result of the Black-Scholes formula and the observed option price. The implied volatility is the parameter value that minimizes this difference.

Once you have completed the BlackScholesCallObject and BlackScholesPutObject, pick a stock for which historical data is available (option prices for various strike prices and expiration dates). You should pick a stock that doesn't pay dividends or one has a low dividend rate.

Find the **implied volatilities** for your stock using the BlackScholesCallObject and BlackScholesPutObject and the Newton Raphson solver in the mathUtil package.

Plot the following four graphs:

Put Option:
1) Implied volatilities against strike price for options with the same maturity.
2) Implied volatilities against different maturity dates with the same stock price.

Call Option
3) Implied volatilities against strike price for options with the same maturity.
4) Implied volatilities against different maturity dates with the same stock price.

## Handin Procedure:

Pittsburgh/NY:  FTP the modified source files and a document (txt, rtf, or doc) containing your answers to the written questions and your plots to the course handin directory (where *userid* is your andrew user id):

/afs/andrew/course/46/935/handin/*userid*/

London:  Email the modified source files and a document (txt, rtf, or doc) containing your answers and plots to jeffreys@andrew.cmu.edu **AND** pavani@andrew.cmu.edu.