CS100M Fall 2005 Project 5 due Thursday 11/10 at 6pm

Submit your three files `Buyer.java`, `PutOption.java`, and `CallOption.java` on-line in CMS under Project 5 before the project deadline. For java code, be careful to submit the .java file, not the .class file. Both correctness and good programming style contribute to your project score.

You must work either on your own or with one partner. You may discuss background issues and general solution strategies with others, but the project you submit must be the work of just you (and your partner). If you work with a partner, you and your partner must register as a group in CMS and submit your work as a group.

Important note on using DrJava

Turn off the file backup feature of DrJava! Several students had problem with submitting the backup file instead of the final version in Project 4. Go to menu item **Edit-->Preferences**, choose the last category, "**Miscellaneous**," then **un**check the box for "**Keep Emacs-style Backup Files**."

Objectives and Background

In this project, you will learn how to use classes and objects and how to develop and test code incrementally—one class (or even one method) at a time. You will implement the buying of stock options for different people on a "trading floor." In finance, stock options are defined as contracts between two parties in which one party obtains the right, but not the obligation, to buy or sell the underlying stock at a fixed price. Because options derive their value from other assets (stocks in this case) they are classified as derivative assets. For example, the company you work for in the future may offer you stock options.

There are two kinds of options – **Calls** and **Puts**. Call options are contracts giving the option holder the right to buy stock, while Put options, conversely, entitle the holder to sell stock. The price you pay for Call and Put options is called a premium. Depending on the stock price, the strike price, the volatility of the stock, the time until expiration, and the risk free interest rate of the market, the premium of each option can vary dramatically.

Call Option Example:
Suppose XYZ Company is currently trading at $95. A Call buyer pays a premium of $15 per share to *buy* 200 shares of XYZ for $100 within 6 months. The premium cost to the buyer is $15*$200 = $3000. The fixed price, in this case $100, is known as the strike price. The Call buyer can then buy the stock for $100 any time before the expiration of the option. If the stock does not go *above* $100 within 6 months, then there is no benefit to buying so the Call buyer never exercises the option and the profit is –$3000 from paying the premium. If the stock price rises to $150 when the Call buyer exercises the option, then the Call buyer would make ($150 – $100)*200 =$10000. Since the Call buyer has paid $3000 in premium, the net profit is $10000–$3000=$7000.

Put Option Example:
Suppose XYZ Company is currently trading at $105. A Put buyer pays a premium of $15 per share to *sell* 200 shares of XYZ for $100 within 6 months. The strike price is again $100. The Put buyer can then sell the stock for $100 any time before the expiration of the option. If the stock does not go *below* $100 within 6 months, then there is no benefit to selling so the buyer never exercises the option and the

profit is –$15*200= –$3000 from paying the premium. If the stock price lowers to $50 when the Put buyer exercises the option, the Put Buyer would have a net profit of ($100 – $50)*200 – $3000 = $7000.

Figure 1 shows the graphical representation of Call and Put options.

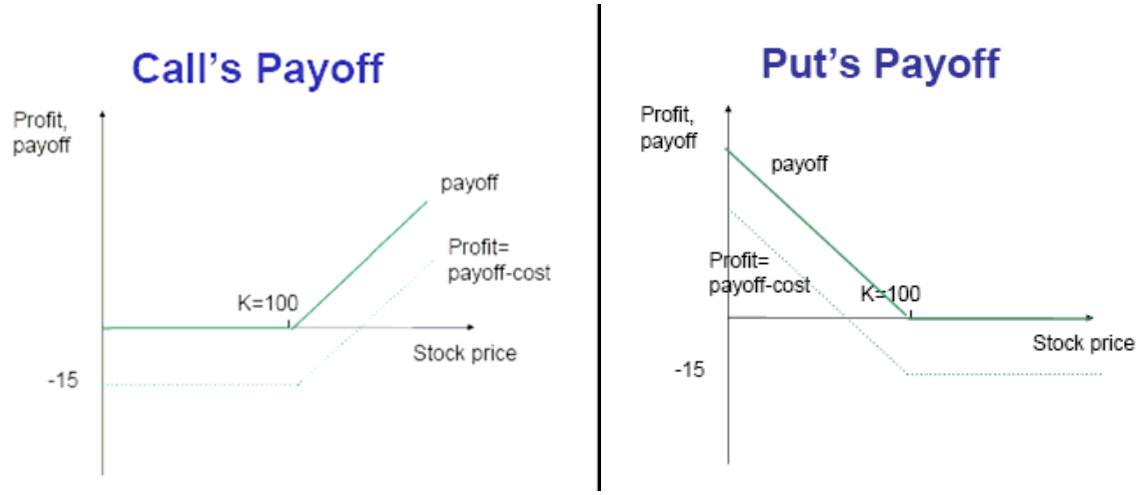

Figure 1: Call and Put payoff graphs where the premium is –$15 and strike price, K, is $100.

Premium Pricing:

The premiums paid for the XYZ stock options in the previous examples did not come out of a hat. In fact, there exists a standard method for calculating an option's premium called the Black Scholes Model. The Black-Scholes model, for which Fischer Black, Myron Scholes and Robert Merton were awarded the Nobel Prize in Economics, is a tool for pricing stock options. The use of the Black-Scholes model and formula is pervasive in financial markets.

The model takes as input the current stock price, length of time until the option expires, an estimate of future volatility known as implied volatility, and the risk free rate of return, generally defined as the interest rate of short term US treasury notes. The main theory behind this model is that an option's price is largely determined by how volatile the underlying stock is and how long the option lasts before it expires.

Black-Scholes Model

Call Option Price: $c = s\Phi(d_1) - xe^{-rt}\Phi(d_2)$ Put Option Price: $p = xe^{-rt}\Phi(-d_2) - s\Phi(-d_1)$

where $d_1 = \frac{log(s/x) + (r + \sigma^2/2)t}{\sigma\sqrt{t}}$ and $d_2 = d_1 - \sigma\sqrt{t}$

`log` = The natural logarithm (base e)
`s` = The price of the underlying stock
`x` = The strike price
`r` = The continuously compounded risk free interest rate
`t` = The time in *years* until the expiration of the option

σ = The implied volatility for the underlying stock
Φ = The standard normal cumulative distribution function.

The class specifications in later sections will explain further the Call and Put options and the calculation of their premiums.

Setup

This project will involve implementing functions in four classes: `Buyer`, `PutOption`, `CallOption`, and `TradingFloor`. The skeletons for the code have been provided for you and are available on the course website. You will complete these classes according to the specifications. You must not change any given variable declarations or given method headers. You may implement additional private methods if you wish, but do not implement additional instance variables, public methods, or classes. Also, do not use arrays—we will use arrays in the last project. Work on the classes one at a time—test each class before moving on to work on the next one. Incremental testing will end up saving you time. The given `Statistics` class is for your use—no implementation necessary.

The starting point of the program is the main method in class `TradingFloor` where a `Buyer`, `PutOption`s, and `CallOption`s are created. You will behave as the Buyer, who can purchase at most one Call and one Put at a time. If the Call or Put held by the Buyer has been exercised, then the Buyer is allowed to purchase another option. The given code in `TradingFloor` simulates the activities of the Buyer and shows an updated price of all the options after each action by the Buyer. See the example output at the end of this document.

Now look at the skeleton code we have provided. We have declared all the instance variables and constants and you will need to fill in the method bodies. Notice that the method bodies are empty except for return statements that match the return type of the methods. We put these return statements in so that the entire program will compile, but of course the logic of the program is incorrect because the real code of the method bodies is missing. You will need to change these return statements as you implement the methods.

First, let's make sure that you can compile the program. Download all the files, including `Statistics.java`, and compile them. Now you can run the `main` method of class `TradingFloor`. Below is a sample starting output where buyer M chooses to exit.

```
Trading Floor Open
Please enter buyer's name:
M
Created buyer M with account balance 10000.0

What would null like to do?
1. Buy Call Option
2. Buy Put Option
3. Exercise Call Option
4. Exercise Put Option
5. Wait for Changes in Stock Prices
6. Exit Program
6
Trading Floor Closed
```

This step ensures that you have code that correctly compiles and executes. From this point on, work on the individual classes one at a time. Compile your code often so that you can identify errors soon after they're introduced. As you work on the individual classes, you probably won't execute the entire program, but still, keep compiling… Since we have given you code that compile to start with, we expect that your submitted program files will at a minimum compile (even if the logic isn't entirely correctly).

The CallOption Class (CallOption.java)

Variables:

```
private String symbol          // Symbol (company name) for the stock option
private double stockPrice      // Current market price of the option
private double strikePrice     // Fixed price at which the Buyer is allowed to buy stocks
private double premium         // Premium (cost to purchase the option) for one share
private int validityPeriod     // Total number of days for which the option is invalid
private int numberOfShares     // Number of shares this option contains
private double volatility      // Standard deviation of stock returns
```

Constructor:

```
public CallOption( String symbol,
                   double strikePrice,
                   double volatility,
                   int validityPeriod,
                   int numberOfShares,
                   double stockPrice)
```

The constructor assigns values to all the fields. To set the `premium` field, see method `calculatePremium` below.

Methods:

```
public double getStockPrice()          // Retrieves current stock price
public void setStockPrice(double s)    // Assigns current stock price
public double getStrikePrice()         // Retrieves option's strike price
public int getShares()                 // Retrieves number of shares
public double getVolatility()          // Retrieves option's volatility
public int getValidityPeriod()         // Retrieves days left in contract
```

```
public void calculatePremium()
```

This method calculates the premium per share according to the Black Scholes Model and assigns the value to the premium field. Refer to the formulae for the Black Scholes Model on page 2 in order to implement this method. Note that the risk free interest rate is a constant in the `TradingFloor` class. To take the natural logarithm, use the method `Math.log`. When determining the normal probability distribution, use the given `Statistics` class. For example, the Java expression to calculate Φ(d1) is `Statistics.normalProbability(d1)`.

```
public String toString()
```

Returns the string representation of this CallOption. It must return at least the symbol, stockPrice, and strikePrice. For the prices, show only 2 decimal places—look up how to use class `NumberFormat` in the API (or you can refer to the text book, Sec 2.2).

Testing: You should be testing your methods as you develop this class! How? In the main method of a new Test class (or DrJava's Interaction Pane), instantiate a CallOption object and call its methods. Print some of the field values and/or calculated results to see if they are what they're supposed to be. Make sure that your CallOption class is working before moving on to the next class.

The PutOption Class (PutOption.java)

This class has the same definition as the CallOption class except that the premium is calculated differently. Implement this class.

The Buyer Class (Buyer.java)

```
Variables:
      private String name            // Name of the Buyer
      private double account         // Amount of money the Buyer has
      private CallOption callOp      // This holds the buyer's CallOption
      private PutOption putOp        // This holds the buyer's PutOption
Constructor:
      public Buyer(String name, double initialAcctBalance)
```

Set the Buyer's name and initial account balance. Initially the Buyer has not bought any options yet.

```
Methods:
      public String getName()        // Retrieves this Buyer's name
      public double getAccount()     // Retrieves the account balance

      public void buyCallOption(CallOption newCallOption)
```

This method allows the Buyer to purchase the newCallOption if the Buyer doesn't already own a CallOption and has enough money to pay the premium (for all shares). Besides updating the relevant fields, this method also displays a message stating the name and the premium (per share) of the newCallOption bought or a message saying why the Buyer cannot purchase the newCallOption.

```
      public void buyPutOption(PutOption newPutOption)
```

This method allows the Buyer to purchase the newPutOption if the Buyer doesn't already own a PutOption and has enough money to pay the premium (for all shares). Besides updating the relevant fields, this method also displays a message stating the name and the premium (per share) of the newPutOption bought or a message saying that the Buyer cannot purchase the newPutOption.

```
      public double exerciseCallOption()
```

This method calculates and returns the payout amount when the Buyer exercises its CallOption. The payout is the maximum of

```
(0, stockPrice-strikePrice)*numberOfShares.
```

All relevant fields should be updated. Print a message stating the payout amount and the Buyer's new account balance. If this Buyer cannot exercise a Call (e.g., the Buyer doesn't own a CallOption), print a message saying so.

```
      public double exercisePutOption()
```

This method calculates and returns the payout amount when the Buyer exercises its PutOption. The payout is the maximum of

`(0, strikePrice-stockPrice)*numberOfShares.`

All relevant fields should be updated. Print a message stating the payout amount and the Buyer's new account balance. If this Buyer cannot exercise a Put (e.g., the Buyer doesn't own a PutOption), print a message saying so.

```
public String toString()
```

Returns the string representation of the buyer. This method should return the Buyer's name and the details of the owned options, if any.

Test the methods in this class! Go to the main method of the Test class that you should have created from testing the CallOption and PutOption classes and instantiate a Buyer object. Be sure to test the buy option and exercise option methods. Have you updated all the relevant fields? E.g., account, CallOp, PutOp. Are the printed messages correct?

The TradingFloor Class (TradingFloor.java)

This class is given. The TradingFloor class is the main entrance to the program. We have created three Put and 3 Call options for a Buyer, i.e., the user, to buy or exercise options. The Buyer has 5 options in the Trading Floor:

1. Buy a Call Option
   A list of available Call Options and their prices will be displayed
2. Buy a Put Option
   A list of available Put Options and their prices will be displayed.
3. Exercise a Call Option
   If the Buyer owns a Call Option, it will be exercised. The new account balance and the payout for the Call will be displayed.
4. Exercise a Put Option
   If the Buyer owns a Put Option, it will be exercised. The new account balance and the payout for the Put will be displayed.
5. Wait for Changes in Option Prices
   Here the price of every Call and Put Option will be updated according to the current stock price and the option's volatility.
6. Exit Program

You don't need to implement this class, but you can experiment with the different stock values or change to some different stocks if you wish. However, do not change the functionality of this class.

```
Variable:
      public static final double RISK_FREE_RATE = 0.1;
Methods:
      public static void main (String[] args)
```

This is the entrance point to the entire program and where the buying or exercising of options occurs.

```
      private static void updateStockPrices(PutOption p1, PutOption p2,
                                            PutOption p3,CallOption c1,
                                            CallOption c2, CallOption c3)
```

This method updates the stock price by generating a random number between 1 and the current stock price, multiplying it by the stock's volatility, and randomly adding or subtracting that value from the current stock price.

```
private static void printMenu(Buyer b)
```
This method prints the menu of options for the user.

```
private static int getIntInput(Scanner keyboard)
```
This is a helper method to read the input values from the user.

The Statistics Class (Statistics.java)

Do not change anything in this file!

```
Method:
      public static double normalProbability(double z)
            This method returns the probability that the standardized normal
            variable Z (mean = 0, standard deviation = 1) is less than z.
```

Congratulations, you have completed this options trading project. Run it and play with the functionality! Computational finance is one of the many fields that computer scientists are concerned about. Most big banks have entire departments that deal with the modeling of stock performance, trends, and investment strategies. Such models tend to be large and computationally intensive (due to the complexity of the model, amount of data and variables, and probabilistic input and statistical prediction). Courses in Computer Science that explore these challenges include CS322 Scientific Computing, CS421 Numerical Analysis, and CS522 Computational Methods for Finance.

Submission Instructions

Your submission should include three source files: `Buyer.java`, `CallOption.java`, `PutOption.java`.
Please ensure that all the files in your program compile correctly before submitting them. Programs that do not compile will receive a significant penalty.

Sample Output: user-entered values appear in *italics*.

```
Trading Floor Open
Please enter buyer's name:
M
Created buyer M with account balance $10000.0
Buyer M owns nothing.

What would M like to do?
1. Buy Call Option
2. Buy Put Option
3. Exercise Call Option
4. Exercise Put Option
5. Wait for Changes in Option Prices
6. Exit Program
1
```

```
Which Call would you like to buy?
1. GOOG Stock: $385.95 Strike: $360.0 Premium: $60.21
2. APPL Stock: $61.85 Strike: $60.0 Premium: $39.78
3. MSFT Stock: $26.44 Strike: $20.0 Premium: $25.44
3
M bought $25.44 call option:
       MSFT Stock: $26.44 Strike: $20.0 Premium: $25.44
M's account balance: $7,455.57

Current stock prices:
EBAY Stock: $40.5 Strike: $30 Premium: $0
AMZN Stock: $40.45 Strike: $50 Premium: $7.62
PIXR Stock: $51.48 Strike: $50 Premium: $0
GOOG Stock: $384.53 Strike: $360.0 Premium: $60.21
APPL Stock: $60.8 Strike: $60.0 Premium: $39.78
MSFT Stock: $25.4 Strike: $20.0 Premium: $25.44

Buyer M owns
 CALL OPTION:
       MSFT Stock: $25.4 Strike: $20.0 Premium: $25.44


What would M like to do?
1. Buy Call Option
2. Buy Put Option
3. Exercise Call Option
4. Exercise Put Option
5. Wait for Changes in Option Prices
6. Exit Program
5

Current stock prices:
EBAY Stock: $39.36 Strike: $30 Premium: $0
AMZN Stock: $39.38 Strike: $50 Premium: $7.62
PIXR Stock: $50.22 Strike: $50 Premium: $0
GOOG Stock: $378.18 Strike: $360.0 Premium: $60.21
APPL Stock: $59.67 Strike: $60.0 Premium: $39.78
MSFT Stock: $24.18 Strike: $20.0 Premium: $25.44

Buyer M owns
 CALL OPTION:
       MSFT Stock: $24.18 Strike: $20.0 Premium: $25.44


What would M like to do?
1. Buy Call Option
2. Buy Put Option
3. Exercise Call Option
4. Exercise Put Option
5. Wait for Changes in Option Prices
6. Exit Program
4
There is no Put Option to exercise

Current stock prices:
EBAY Stock: $38.35 Strike: $30 Premium: $0
AMZN Stock: $37.82 Strike: $50 Premium: $7.62
PIXR Stock: $49.18 Strike: $50 Premium: $0
GOOG Stock: $373.82 Strike: $360.0 Premium: $60.21
APPL Stock: $58.59 Strike: $60.0 Premium: $39.78
MSFT Stock: $23.15 Strike: $20.0 Premium: $25.44

Buyer M owns
 CALL OPTION:
       MSFT Stock: $23.15 Strike: $20.0 Premium: $25.44
```

```
What would M like to do?
1. Buy Call Option
2. Buy Put Option
3. Exercise Call Option
4. Exercise Put Option
5. Wait for Changes in Option Prices
6. Exit Program
2
Which Put would you like to buy?
1. EBAY Stock: $38.35 Strike: $30 Premium: $0
2. AMZN Stock: $37.82 Strike: $50 Premium: $7.62
3. PIXR Stock: $49.18 Strike: $50 Premium: $0
2
M bought $7.62 put option:
       AMZN Stock: $37.82 Strike: $50 Premium: $7.62
M's account balance: $6,693.09

Current stock prices:
EBAY Stock: $39.46 Strike: $30 Premium: $0
AMZN Stock: $38.96 Strike: $50 Premium: $7.62
PIXR Stock: $50.39 Strike: $50 Premium: $0
GOOG Stock: $376.94 Strike: $360.0 Premium: $60.21
APPL Stock: $60.2 Strike: $60.0 Premium: $39.78
MSFT Stock: $24.21 Strike: $20.0 Premium: $25.44

Buyer M owns
 CALL OPTION:
       MSFT Stock: $24.21 Strike: $20.0 Premium: $25.44
 PUT OPTION:
       AMZN Stock: $38.96 Strike: $50 Premium: $7.62


What would M like to do?
1. Buy Call Option
2. Buy Put Option
3. Exercise Call Option
4. Exercise Put Option
5. Wait for Changes in Option Prices
6. Exit Program
2
Which Put would you like to buy?
1. EBAY Stock: $39.46 Strike: $30 Premium: $0
2. AMZN Stock: $38.96 Strike: $50 Premium: $7.62
3. PIXR Stock: $50.39 Strike: $50 Premium: $0
3
Unable to buy option. M already owns one Put Option.

Current stock prices:
EBAY Stock: $38.25 Strike: $30 Premium: $0
AMZN Stock: $37.51 Strike: $50 Premium: $7.62
PIXR Stock: $49.2 Strike: $50 Premium: $0
GOOG Stock: $370.87 Strike: $360.0 Premium: $60.21
APPL Stock: $58.68 Strike: $60.0 Premium: $39.78
MSFT Stock: $23.06 Strike: $20.0 Premium: $25.44

Buyer M owns
 CALL OPTION:
       MSFT Stock: $23.06 Strike: $20.0 Premium: $25.44
 PUT OPTION:
       AMZN Stock: $37.51 Strike: $50 Premium: $7.62


What would M like to do?
1. Buy Call Option
2. Buy Put Option
```

```
3. Exercise Call Option
4. Exercise Put Option
5. Wait for Changes in Option Prices
6. Exit Program
4
AMZN Stock: $37.51 Strike: $50 Premium: $7.62
M exercised Put for payout of $1,248.58.
 New account balance: 7,941.68

Current stock prices:
EBAY Stock: $39.46 Strike: $30 Premium: $0
AMZN Stock: $39.01 Strike: $50 Premium: $7.62
PIXR Stock: $50.47 Strike: $50 Premium: $0
GOOG Stock: $373.85 Strike: $360.0 Premium: $60.21
APPL Stock: $61.3 Strike: $60.0 Premium: $39.78
MSFT Stock: $24.11 Strike: $20.0 Premium: $25.44

Buyer M owns
 CALL OPTION:
       MSFT Stock: $24.11 Strike: $20.0 Premium: $25.44


What would M like to do?
1. Buy Call Option
2. Buy Put Option
3. Exercise Call Option
4. Exercise Put Option
5. Wait for Changes in Option Prices
6. Exit Program
3
M exercised Call for payout of $411.34.
 New account balance: 8,353.02

Current stock prices:
EBAY Stock: $38.44 Strike: $30 Premium: $0
AMZN Stock: $37.56 Strike: $50 Premium: $7.62
PIXR Stock: $49.42 Strike: $50 Premium: $0
GOOG Stock: $368.62 Strike: $360.0 Premium: $60.21
APPL Stock: $58.4 Strike: $60.0 Premium: $39.78
MSFT Stock: $22.87 Strike: $20.0 Premium: $25.44

Buyer M owns nothing.

What would M like to do?
1. Buy Call Option
2. Buy Put Option
3. Exercise Call Option
4. Exercise Put Option
5. Wait for Changes in Option Prices
6. Exit Program
2
Which Put would you like to buy?
1. EBAY Stock: $38.44 Strike: $30 Premium: $0
2. AMZN Stock: $37.56 Strike: $50 Premium: $7.62
3. PIXR Stock: $49.42 Strike: $50 Premium: $0
2
M bought $7.62 put option:
       AMZN Stock: $37.56 Strike: $50 Premium: $7.62
M's account balance: $7,590.54

Current stock prices:
EBAY Stock: $37.11 Strike: $30 Premium: $0
AMZN Stock: $35.89 Strike: $50 Premium: $7.62
PIXR Stock: $48.16 Strike: $50 Premium: $0
GOOG Stock: $361.78 Strike: $360.0 Premium: $60.21
APPL Stock: $57.28 Strike: $60.0 Premium: $39.78
```

```
MSFT Stock: $21.58 Strike: $20.0 Premium: $25.44

Buyer M owns
 PUT OPTION:
       AMZN Stock: $35.89 Strike: $50 Premium: $7.62


What would M like to do?
1. Buy Call Option
2. Buy Put Option
3. Exercise Call Option
4. Exercise Put Option
5. Wait for Changes in Option Prices
6. Exit Program
6
Trading Floor Closed
```