CS110 Assignment #3

Due *Friday*, February 6th (Although the assignment is due after the exam, the knowledge required to complete this assignment will be tested. I advise having the majority of all functionality complete before the exam).

READ THIS DOCUMENT THOROUGHLY!

You will have your first exam on Wednesday, February 4th during lab time (location to be announced on BB). Chapters 1-5 in the textbook will be covered. Tuesday's (2/3) lecture material, Chapter 6 – A First Look at Classes - will not be covered on the exam; however, it is one of the most important lectures of the semester. I strongly encourage attendance. Due to the exam, there will be no lecture on Thursday (2/5).

There is a lot of work to be done on this assignment (read to the end to see that you have 5 programs to write). Do not delay starting! The assignment is due on a Friday, there are no TA hours on Friday, seek help early. The full TA schedule is available on Blackboard. There is a large block of time available on Sundays (1-7pm)!

REMINDER: Any work you submit for this assignment should be authored entirely by yourself. Assistance is permitted from the instructor or teaching assistants only. All submitted programming assignments will be subject to originality verification through software designed and used for the Measure Of Software Similarity (MOSS).

The Assignment

*For this, and all future assignments, submit your assignment as one compressed **(.zip)** file. If you are not sure how to do this, please see a TA in lab!*

1. Complete File Letter Counter programming challenge (4-6). All user input/output will use JOptionPane.
   Note – the character specified is case sensitive (i.e., 'a' and 'A' will return different results).

2. Complete Uppercase File Converter programming challenge (4-15). Input/Output will use JOptionPane.

3. Write a program that generates a random number and asks the user to guess what the number is. If the user's guess is incorrect, the program should display "too high, try again" or "too low, try again". Repeat until the user has guessed correctly or exceeded the number of attempts. The program should issue a final message indicating the number was guessed and the number of attempts it took, or the number that was not guessed.

   This program will run at the command prompt (using System.out.println and Scanner).

   The upper limit of number to be guessed and the max number of guesses should be constant values. I should be able to change those two constants and the rest of the program will operate properly (including output statements).

```
Sample program run #1:

I'm thinking of a number between 1 and 100
You have 5 tries to guess it
Guess 1: 50
Sorry, that's too high.
Guess again: 25
No, that's too low.
Guess again: 35
No, that's too low.
Guess again: 40
No, that's too low.
Guess again: 45
You are out of guesses.
I was thinking of the number 49.

Sample program run #2:

I'm thinking of a number between 1 and 100
You have 5 tries to guess it
Guess 1: 50
Sorry, that's too high.
Guess again: 25
Sorry, that's too high.
Guess again: 15
Congratulations! You guessed it!
You got it right in 3 guesses.
```

4. Complete Conversion Program programming challenge (5-8). This program will run at the command prompt (using System.out.println and Scanner). No "magic numbers"!

5. Complete Present Value programming challenge (5-16). The formula assumes the interest rate is in decimal form (.025 for 2.5%), however, the user should enter it as the true percentage (2.5%). This program will run at the command prompt (using System.out.println and Scanner).

   A sample run:

```
What is the desired future value? 100000
What is annual interest rate (%)? 2.5
For how many years? 10
You need to invest $78,119.84
```

Remember programming style!

- Indentation, appropriately named variables (follow naming conventions), comments (header, inline comments)
- No break or continue with the exception of within a switch statement
- All variables must be declared locally (within a function). Named constants may be declared at the class level.
- Your programs should be well-documented. Don't forget the @param and @return documentation on methods.