**COSC 1047 – Winter 2017 – Assignment 4  
See D2L For Due Date**

This assignment is designed to give you some experience writing Java GUIs. This assignment is done using Eclipse. Import the .zip file that is provided on D2L and when it is complete, export it again as a .zip. All submissions should be done through D2L. You can submit and resubmit as many times as you want up to the due date and time. If you have any problems submitting, email me – [alangille@cs.laurentian.ca](mailto:alangille@cs.laurentian.ca) - in a timely manner. **Make sure your name, student number and a brief description of your program appear at the top of each file in comments. Also make sure that you use comments to clarify your code for marking. *Failure to do either of these tasks may result in a deduction of marks.***

***Submit solutions to questions 3 OR 4 and 5 only. But DO ALL questions to be prepared for the tests and exam.***

***Remember if you work in partners to include all names in the source code. The person who is left out will receive zero.***

Note: It is imperative that you import the a4-w16.zip file from D2L properly or you won’t be able to work in the proper project environment. Although the screen shots are a bit outdate these instructions are correct:

<http://agile.csc.ncsu.edu/SEMaterials/tutorials/import_export/>

Grading:

For the submitted questions the grading is as follows:

Proper documentation, formatting, comments and Java Docs where requested: 5 marks

Program compiles properly and runs: 5 marks

Program runs properly for most of the examples and requirements of the question: 5 marks

Program runs properly for all of the examples and requirements of the question: 5 marks

Extra work where applicable: 5 marks

Small non-critical errors and suggestions for improving code will be given as comments and should be addressed in future assignments. Major errors and failure to follow the question guidelines will result in deductions per the scheme listed above.

1. Write a GUI for computing the statistics of numbers input from the user. The GUI should have a text field and a button for adding floating point numbers. Each time a number is entered have a label show the current max, min and average. Have your GUI accept numbers whether the button is pressed or the enter-key is pressed in the data field.
2. Write a GUI with three text fields, one for the initial amount of a savings account, the annual interest rate and a number of years. Add a button “Calculate” and a read-only text area to display the balance per year. Show both the balance and the year in a format similar to:

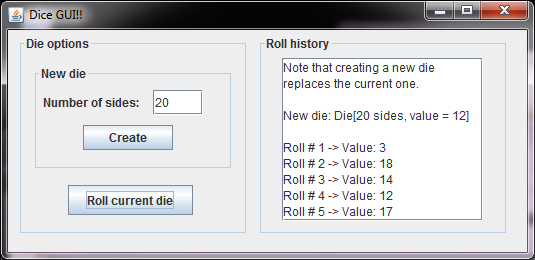
Initial amount: $50.00  
After year 1: $51.00

After year 2: $52.00

…

(Extra) Your GUI should really show an error (JOptionPane?) if any of the fields are negative or are invalid data types like strings.

1. (DieGUI.java) Write a GUI that will allow you to simulate a Die. The GUI should take input from the user for the number of sides of the Die and then it should create the Die accordingly. The GUI should then allow the user to roll the Die. It should display the number of times the Die has been rolled as a history of the numbers that have been rolled. If the user creates a new Die (maybe with a different number of sides) the GUI should reset the history and the number-of-rolls counter. Make sure to use the Die class, do not re-write the die functionality in your GUI.   
     
   \*\*\* Your GUI does not have to look as structured as the one below. This is just a sample GUI.



(Extra) If the user enters invalid data for number of sides for the Die, catch the exception and show and error in a JOptionPane. Keep track, somewhere on your GUI, of the frequency of each Die value that has come up so far.

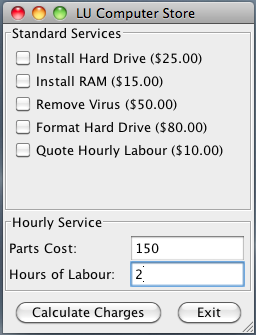
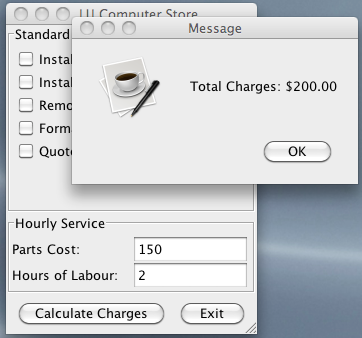
1. (MyStoreGUI.java) The LU Computer Store performs the following standard services (not the prices for the real store):

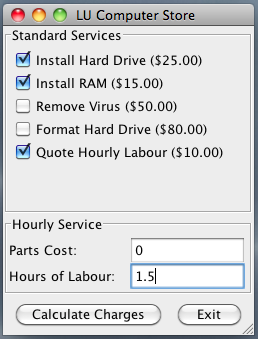
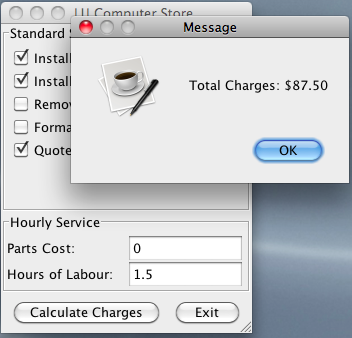
* Install Hard Drive ‑ $25.00
* Install RAM ‑ $15.00
* Virus Removal ‑ $50.00
* Format Hard Drive ‑ $80.00
* Quote Hourly Labour ‑ $10.00

The store also performs other hourly service and charges for parts and labor ($25 per hour). Create a GUI application that displays the total for the customer’s visit to the LU Computer Store. Output should be displayed in a JOptionPane. Make use of JPanels in your GUI and use at least one GridLayout.

Catch the appropriate exception if data is entered in either of the two text fields that cannot be properly converted into a double value – i.e., a string like “apples” or an improperly-formed number like 2.9.1. When this exception is caught display a new message dialogue that invalid input has been received and do not display the charges. The user should be able to go back and correct the error.

Fancy border text? Yes please!

** **

** **

1. (**Recursion.java**) Write a java program that has three static recursive methods and a main method. In your main method (5 marks) prompt for user input and display the results of each of your recursive methods as shown in the sample output below.

a) Write a method that uses recursion to compute the value of , where a and n are both arguments to the method. If *n* = 0, the method should return 1 as . If n = 1, the method should return *a* as . If *n* is any other number … that’s for you to determine but remember, .

b) Write a method that uses recursion to return the reverse of a String that is passed to the method as an argument. Hint: For base cases, consider a string that has 1 or fewer characters…how much work is there to reverse them? Otherwise a reversed string is the last letter of the original string plus the reverse of the rest. Try it out on paper first.

c) Write a recursive method that determines the number of digits in an integer, n. Hint: If n < 10, there is one digit. Otherwise, it has one more digit than n / 10.

For these questions and with recursion in general, the trick is to determine 1) the base case(s) that will cause the function to simply return and end the recursion and 2) the recursive case(s) that will cause the function to call itself with a smaller version of the same problem. All input and output should take place in your main method. \*\*\* Don’t forget to “eat the new line” if you use your scanner between numbers and strings.

**java Recursion**

Enter two numbers, base then exponent: **2 0**

Result: 1

Enter a string to reverse: **apples**

Result: selppa

Enter a number: **123**

Number of digits: 3

**java Recursion**

Enter two numbers, base then exponent: **100 1**

Result: 100

Enter a string to reverse: **Z**

Result: Z

Enter a number: **6**

Number of digits: 1

**java Recursion**

Enter two numbers, base then exponent: **2 12**

Result: 4096

Enter a string to reverse: **i luv cosc!**

Result: !csoc vul i

Enter a number: **1234567890**

Number of digits: 10