CS1073 Assignment #10 - Fall 2020

Submission Deadline: Friday, November 27th before 12:00 NOON (Atlantic Daylight Time Zone) in the Assignment 10 dropbox in Desire2Learn. (Read the submission instructions at the end of this document carefully).

The purpose of this assignment is:

- to work with loops and string manipulation
- to gain a bit more practice with JavaFX GUIs and event-driven programming

This assignment is to be done individually. If you have questions, direct them to a tutor/assistant during a help session in the "Faculty of Computer Science Student Success Centre" team or to your course instructor.

---

1. Converter

For this assignment you will write a class called **Converter** that contains two static methods, **hex2Decimal** and **english2encrypted**. The **Converter** class will be structured like the Math class. We will call the static **hex2Decimal** and **english2encrypted** methods directly (using the class name); we will not create **Converter** objects. Include Javadoc comments for the **Converter** class (description and @author tag) and for each method.

A. Hex to Decimal Converter

While we as humans are used to working with the decimal number system (base 10), computers use binary (base 2). Another representation that is convenient for representing large binary numbers is hexadecimal (base 16). To go beyond the 10 decimal digits (0 through 9), the hexadecimal representation uses the characters **a** through **f** (upper or lower case) to represent the digits whose value is 10 through 15. For example, 5a in hexadecimal is equivalent to 90 in decimal.

In the **Converter** class write the static method, **hex2Decimal(String hex)**. The parameter for this method is a nonnegative hexadecimal number represented by a string. The method returns the decimal equivalent for the hexadecimal parameter, as a **long** integer. If the hexadecimal string is invalid (contains characters other than **'a'-'f'**, **'A'-'F'**, **'0'-'9'**), the method returns **-1**.

Convert from a hexadecimal (h) string with up to 8 digits to decimal (d) using the following formula:

$$d = h_0 + h_1 \times 16 + h_2 \times 16^2 + \cdots + h_7 \times 16^7$$

where $h_0, h_1, h_2, \ldots h_7$ are the hexadecimal digits in order of increasing significance. For example, f541 is: 1 + 4×16 + 5×16^2 + 15×16^3 = 62785 in decimal.

Side note: You may NOT use the **decode** method of the **Long** wrapper class instead of this formula.

B. Secret Language Converter

In the **Converter** class write the static method, **english2encrypted(String english)**. The parameter for this method is a word or sentence represented by a string. The word or sentence should only contain letters (no digits, punctuation, or other special characters); however, it may contain both upper- and lower-case letters. If you want to include a number you must write out the word that represents the number (ie: 8 must be written out as "eight"). The method returns the encrypted equivalent for the English word or sentence parameter, as a string.

The rules for encrypting the English words are as follows:

- The encrypted words will all be in upper-case letters.
- If a word has 2 or more letters, switch the first and last letter.
- Replace all E's in the English words with A's, and replace A's in the English words with E's.
- Replace all O's with I's, and replace I's with O's.
- Replace all U's with Y's, and replace Y's with U's.

Examples:

"duck" becomes "KYCD"

"Feature" becomes "AAETYRF"

"Hello world" becomes "IALLH DIRLW"

"my hidden message" becomes "UM NODDAH AASSEGM"

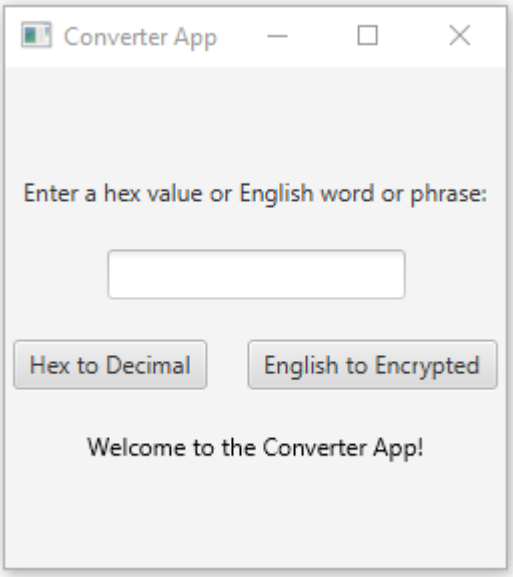
"I got it" becomes "O TIG TO"

Note: You do not need to check for invalid input for the encryption. If the string parameter is invalid (i.e. it contains characters other than letters or spaces), you should not do anything special; the string will be encrypted according to the rules provided.

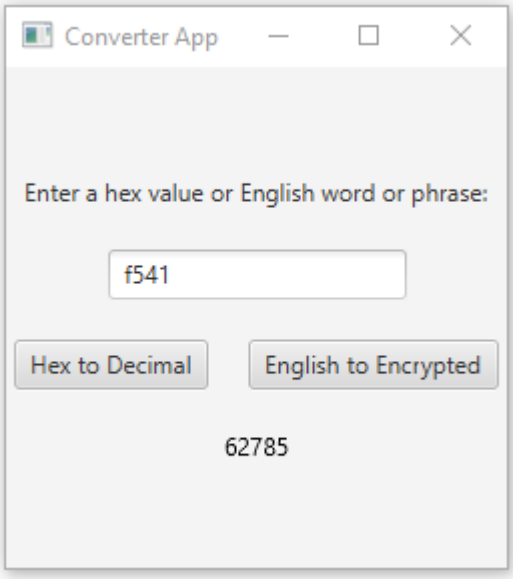
2. Test Driver GUI Application

Create a GUI that can be used to test the methods in your **Converter** class. This GUI must be implemented as a JavaFX application. For invalid hex values the GUI needs to display a message to let the user know their input was invalid. Please follow the sample screenshots shown below.

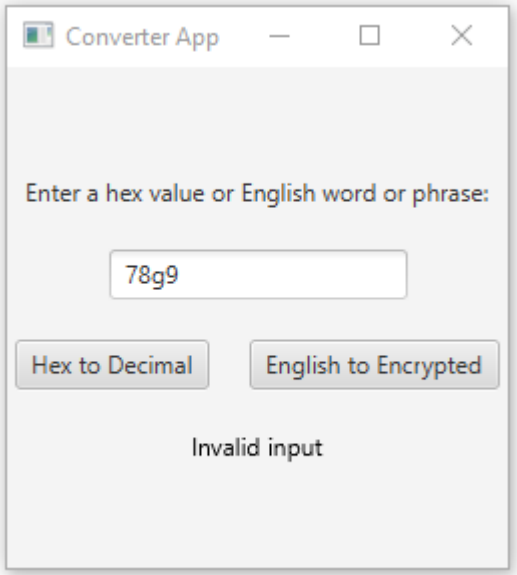
Here is a screenshot showing the application when it is first launched:



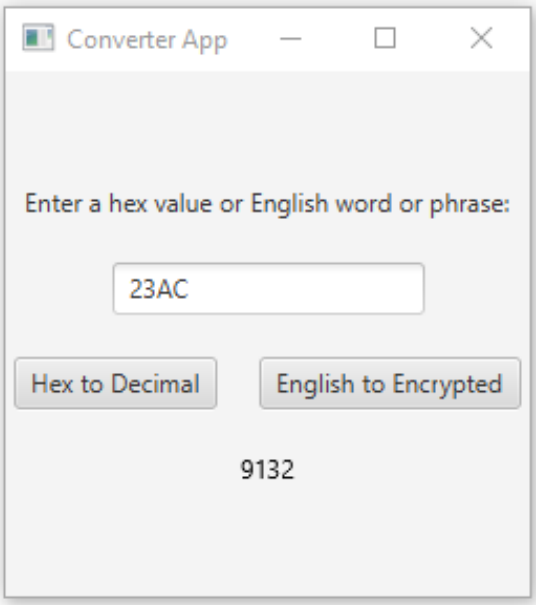
Here is another view after the user has entered a valid hex value and pressed the "Hex to Decimal" button.



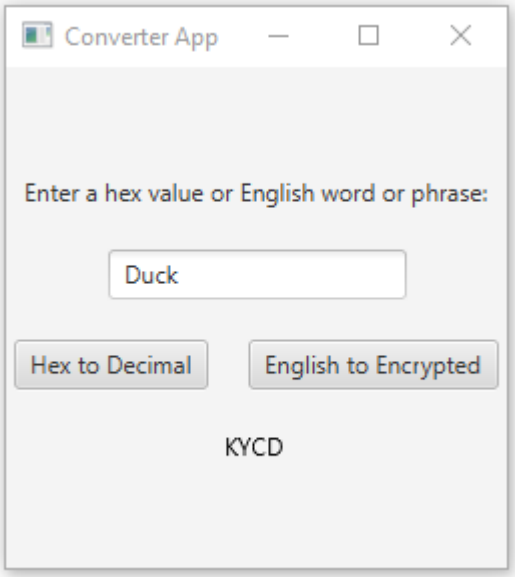
Here is another view after the user has entered an invalid hex value and pressed the "Hex to Decimal" button.



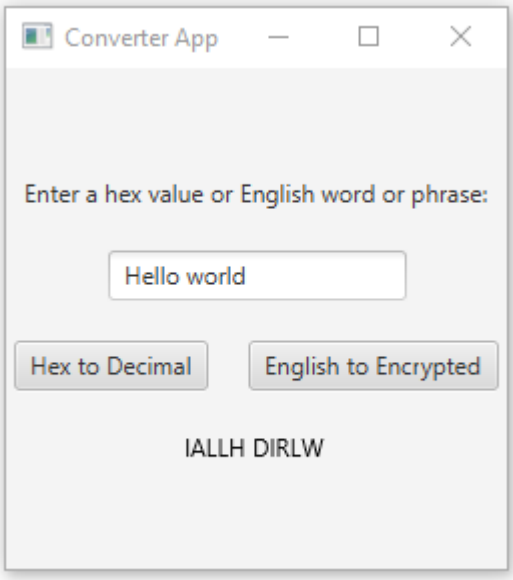
Here is another view after the user has entered a valid hex value and pressed the "Hex to Decimal" button.



Here is another view after the user has entered a word and pressed the "English to Encrypted" button.



Here is another view after the user has entered a phrase and pressed the "English to Encrypted" button.



Aside: You do not need to write full Javadoc comments for the GUI, but please include a Javadoc comment at the start of the class with an @author tag.

Run your program and capture at least 5 screenshots with 5 different sample inputs to submit as sample output. Include at least one invalid input for Hex to Decimal. You should use only valid entries when capturing sample output for the English to Encrypted button.

Submission instructions are on the next page…

Your electronic assignment submission (submitted via Desire2Learn) will consist of two files:

i. a written report. This should begin with a title page; just as we described in Assignment #1, your title page should include: the course (CS 1073), your section (FR01A, FR02A, FR03A or FR04A), the assignment number, your full name, and your UNB student number. That should be followed by two sections, with each part clearly identified with a section heading. Include:

   a. The source code for your Converter class from Question 1 and your JavaFX GUI from Question 2.

   b. At least 5 screenshots to show sample output for the GUI, each with a descriptive label.

   (Aside: Your source code should contain Javadoc comments, however, you do not need to include the .html files.)

   This written report should be prepared using a word processor; we recommend using Microsoft Word (i.e. create a .docx file for your report). Copy & paste your java source code & required output into the report document. Add appropriate headings for each part. Fix up the formatting where necessary, adjusting line breaks & page breaks to ensure that your document is easy to read. Use a monospaced font for your code to maintain proper indentation.) Once the report is complete and you've checked it all over, save the .docx file for your own records, and then **save a second copy in pdf format for submission**. (Note: Be sure to open that file in a pdf viewer to verify that the pdf was generated correctly.) The **SINGLE pdf file** containing your report will be submitted to the appropriate assignment drop box on Desire2Learn. (It is important that you submit a pdf file and NOT the original Word document. This pdf will allow the marker to write comments directly on your work to give you better feedback.)

   Note: Please name this report as follows: **YourName_As10_Report.pdf**

ii. an archive file (**.zip**) that contains your Java source code and the sample output for this assignment. Make sure that your archive includes all .java files (in case the marker wishes to compile & run your code to test it). You should not include the report document or the .class files in your archive. This archive should be submitted as a **single file** to the appropriate drop box on Desire2Learn.

   Note: Please name this archive file as follows:
   YourName_As10_Archive.zip