Project 1: Encryption/Decryption Utility

Introduction:

Encryption is a necessity in today's digital landscape. For this project you will be writing a simple encryption utility for encrypting or decrypting a single message.

You will be encrypting a plaintext message and outputting the ciphertext in hexadecimal format to the console and storing it in a file. Conversely, when decrypting the ciphertext that you read from the file.

Requirements:

The following format should be implemented exactly.

1. When your program starts it will print the following menu:

```
Welcome to My Encryption/Decryption Program
Please enter the letter of your chosen operation:
      a) Encrypt a message
      b) Decrypt a message
      c) Exit
```

   a. Notice that there is an indentation for the menu options. Use a tab character (\t) for this indent.
   b. All strings for output must be stored in a variable.
2. You will take the user's input using *Scanner* as demonstrated in class. The prompt that *input* should display is, "Enter option: "
3. You must validate user input. If the user enters anything other than a, b, or c (these can be upper or lower case) then you must display the error message, "Invalid choice, please try again" then print the menu and input prompt again. Be sure that each prompt is on its own line.
4. You will take the following input using the *Scanner*.
   a. If option "a" is chosen (encrypt), prompt for, and take the message to be encrypted with the prompt, "Input your message: ".
      You will then prompt for a filename in which to store the ciphertext with the prompt "Enter filename:".
   b. If option "b" is chosen (decrypt), you will prompt the user for the filename with, "Enter filename: ". After the user inputs the filename you will open the file and read the first line which should have the ciphertext. You will need to first test if that file exists. If it doesn't print the error message, "That file doesn't exist, Please enter the correct filename or press ctrl-c to exit", then prompt the ussr for the filename again.
5. Next you will take the password as input. Prompt for the password using the prompt, "Input password (must be greater than 8 characters) : "
   a. If the user enters a password less than or equal to a length of 8 , print "Invalid password length. Please try again or press ctrl->c to quit.", and then reprompt for the password input. Be sure that each prompt is on its own line.
6. You then will either encrypt or decrypt using the provided algorithm (below)
7. Your program must be written in one class named "Cipher" which must contain the main method.

Encryption/Decryption Algorithm:

- Iterate over the two strings
    - In the case of encryption, use the plaintext message and the password
    - In the case of decryption, use the ciphertext message (a hex string) and the password
        - Note: the password might be shorter than either the plaintext or the ciphertext
            - Each two hex characters in the ciphertext will have to be converted to an integer value Note: each two digits in the hex string represent a number but they are still just characters in a string. They'll need to be converted to the integer value of the number that they represent using `Integer.parseInt(hex,16)`.
            - E.g., the value 0x41= 'A'. You do not have to create a map to do this. Use the two hex characters to get the char equivalent. Look at the `Integer.toHexString(value)` to get the hex string from an integer value, `Integer.parseInt(hex,16)` to get an integer value from a hex string, `(int) character` to get the integer value of a character, and `(char) int_val` to convert an integer value to a character. Note: you can XOR two chars directly, e.g., `char result = 'A' ^ 'B';`. They needn't be converted to integers first.
    - The second string mentioned above is the password.
- You will encrypt the characters of the message or decrypt the hex-values of the ciphertext by XORing them with the corresponding character of the password.
    - Note, the password is likely shorter than the plaintext or ciphertext so you will have to iterate over the password cyclically. That is, when your loop uses the last character of the password it should start again at the beginning character of the password for the next iteration.
- The *result* is the output of the encrypt operation. You will output it to the console and store it in the file with the filename that the user provided. If this is the encryption operation you will need to output the result as a hexadecimal string beginning with "0x", if this is a decryption operation then you should read the ciphertext (a hex-value string) with the above steps and output it to the user.

Restrictions:

- You must use nano on the remote server, login.cpp.edu (abbott or costello) to write your code
    - Note, be careful when copying the final program to your home computer. If you copy it several times and put it in different locations you may end up with multiple copies. This could cause you to submit the wrong version of your code.
- Your program must be written for Java 8. This is your required JVM for the remainder of this course.
- You must follow the Java coding standards (see the link under “Course Documents” on Blackboard).
- Your program should exist in one file named “Cipher.java”.
- **Important**: if your program fails to compile you will not receive any credit for your work. Be sure to start early on this project and leave plenty of time to test and submit it. Be sure to ask me for help if you need it.
- This is not a team project so do your own work. You can discuss the algorithm but never code.
- Do not use code from any other source except that code which was provided by me--still, do not copy and paste it, rewrite it. **You must create your own code!**
- You will be graded heavily on code quality so take your time in implementing your code using the Java coding standards. Before coding I recommend you create notes on what needs to be done. Once you thoroughly understand the algorithm, then begin to code it. These notes can come in the form of comments in your source code file.

Submission:

- Submit your single Java file named “Cipher.java”. Do not include any other files.
- Use the assignment link posted on Bb to submit your file. It will be located under the assignment folder where you found this file.