CS-207: Programming II
Fall 2016
Northeastern Illinois University
Research Lab: XOR Encryption
Due: Tuesday, 12/06 at 9:00 a.m.

Goal:

The goal of this research lab is to investigate the XOR encryption algorithm and encrypt a file using the XOR technique while using exception handling techniques described in your course. In addition, students will become more familiar with referencing the Java 8 docs when working with new API classes.

The Problem:

You have been provided with a text file that you will eventually encrypt and decrypt using the XOR Encryption Algorithm. This research lab will have significantly less instruction as this will more accurately mimic the research process that you may encounter in a job situation or academic research situation.

Instructions:

- You should work in groups of 2-3 individuals. Groups of more than 3 are **not** permitted.
- Each group should submit ONE lab write-up. It is the responsibility of each group member to ensure that their name is on the write-up.
- The lab write-up should be typed! Type each question (and the question number) followed by your group's answer. **Convert your lab write-up to a .pdf.**
- You should use complete sentences and proper grammar in your write-up. Use spell-check! This counts as part of your grade.
- You should not copy/paste directly from your sources for your answers (this is called plagiarism). Instead, you should re-word the information in your own words.
- Submit the pdf and .java file containing the code for encrypting a file to D2L by the specified due date.
- Each member of the group must turn in a hard-copy of the peer assessment on the day the research lab is due. The peer assessment counts as a significant part of your grade and you will receive a **zero** for that portion of the research lab grade if you do not turn it in.

Part A: Investigating the XOR Encryption Algorithm

Use Google to investigate the XOR encryption algorithm and answer the questions below.

Question #A.1

What is the XOR (exclusive or) operator? Create a truth table for for two bits for the XOR operator.

Question #A.2
The number `4` can be represented in binary by `00110100` and the letter `Q` can be represented in binary by `00110100`. Find the value of `Q XOR 4`.

Question #A.3
Describe how the XOR encryption algorithm works and why it is a robust encryption algorithm.

Part B: Getting Started
Since you will be working with encrypting files and a file that contains a key, it is important to research the necessary Java 8 API classes for working with files.

Question #B.1
You need one specific Java package imported in order to work with `File` objects. What is it? Import this package before your class definition. Each file that you read from needs to be represented by a `File` object.

Question #B.2
Download the `data.txt` file from the NeededFiles.zip file and save it in the same folder as your Java file. Look at the data in this file - do you recognize it? Create a .txt file named `key.txt` that has the following text: `This is a key`

Question #B.3
You will need to be able to read from each of your `File` objects. To do this, you will use the `FileInputStream` class. Why is it necessary to use the `FileInputStream` class instead of the `Scanner` class? Find the constructor for the`FileInputStream` class in the Java 8 docs that takes a `File` object as a parameter. What type/name of the exception does it throw? Why do you need to know this before you create a `FileInputStream` object? Create a `FileInputStream` object for each of your `File` objects. Make sure to handle the `FileNotFoundException`.

Question #B.4
The `read` method of the `FileInputStream` class requires an initialized byte array for a parameter. Create and initialize two byte arrays. How do you know the `length` of the arrays? (Hint: read the Java docs for the `File` class. Remember, in order to declare and create your arrays, you need an `int` - you may find casting helpful!). Make sure to handle any exceptions.

Question #B.5
Read from your `data.txt` and `key.txt` files. Make sure to handle exceptions.

Part C: Implementing the XOR Algorithm
To implement the algorithm, you need to iterate over the byte array created by reading from the `data.txt` file.

Question #C.1
For each byte in the `data.txt` file array, you should find the index of the key byte array by taking the for-loop counter `mod` the length of the key byte array. Then, encrypt the byte: (byte

at the for loop counter) XOR (key_byte at index found using `mod`). Assign the encrypted byte back to the byte array (for the`data.txt` file). What is the XOR operator in Java?

Part D: Writing the Encrypted Data

Question #D.1
You need to write the encrypted byte array to a file using a `FileOutputStream` object. Why is it necessary to use the `FileOutputStream` class instead of the `Scanner` class?

Question #D.2
Run your code and verify that the contents of the encrypted file look like nonsense. Then, re-run your code with the encrypted file (this will decrypt it) to verify that you get your original file back.

Part E: Final Summary
Whenever you complete a project, it is important to assess what you think went well and what you need to improve on.

Question #E.1
What was the most challenging part of this research lab for your group?

Question #E.2
What did your group learn/find the most useful by doing this research lab?

Question #E.3
How was this lab different from the previous labs for your group.

Question #E.4
What was the most fun aspect of doing this research lab?