HW3 – (Super) Caesar Cipher
Assigned: Friday, June 4th, 2010
Due: Friday, June 11th, 2010 before 10:00pm

Overview
In this assignment, you will implement one of the most basic encryption schemes in the field of cryptology, the Caesar Cipher. After implementing an encoder and decoder for this cipher, you will then implement an "interceptor" that will "crack" the Caesar cipher encryption when the key is not known.

CaesarEncode.java

In order to encode a file using the Caesar cipher, you need the file and the key. You should take the name of the file to be encoded and the key as command line arguments to the main method of CaesarEncode. Hence, args[0] will be a String fileName and args[1] will be an int key (aka shift number). You may write this file however you want, but CaesarEncode must produce a file xxxencoded.txt upon the command line call

java CaesarEncode xxx.txt z

where xxx.txt is the original file to be encoded and z is the int key.

Your CaesarEncode class must produce a file containing encoded text. The new file's name will be the same as the original, but with "Encoded" added to it. See examples:

java CaesarEncode story.txt 7
produces a file named storyEncoded.txt

java CaesarEncode dataResults.txt 3
produces a file named dataResultsEncoded.txt

Specific Requirements
- Must take specified data as command line arguments to main method
- Produced text file must be named following the format listed above. For example, if foo.txt will cause fooEncoded.txt to be produced.
- Any and all English text should be able to be encoded. This includes punctuation, numbers, symbols, etc (%, :, 67, *, hello world, etc).

CaesarDecode.java

In order to decode a file encoded by the Caesar cipher, you need the encoded file and the key. As in CaesarEncode, you should take the name of the encoded file and the key as command line arguments to the main method (args[0] will be String fileName and args[1] will be the int key). Write this file however you want, but CaesarDecode must produce a file xxxDecoded.txt upon the command line call
java CaesarDecode xxxEncoded.txt z

where xxxencoded.txt is the file to be decoded (you can assume that it is actually encoded) and z is the int key. To be more concrete, your CaesarDecode should produce a decoded file of the same name as the read in encoded file, with the substring 'Decoded.txt' attached to the original file name less 'Encoded.txt'. The new text file should appear in the folder containing xxxencoded.txt and your Java files.

Specific Requirements
- Must take specified data as arguments to main method
- Produced text file must be of the form xxxDecoded.txt where xxxEncoded.txt is the name of the file passed in.
- The decrypted text should exactly match the original text that was encrypted.

CaesarIntercept.java

Take a moment to think about how you can break a code given only encoded text that you know was encoded using the Caesar cipher. To simplify the problem, we will assume that the encoding was done in the same fashion as in the CaesarEncode file you wrote. You cannot just run your CaesarDecode, however, because you do not know the key. Come up with a solution, and implement it in CaesarIntercept.java. CaesarIntercept should take the name of the encoded file as an argument to the main method; hence, args[0] will be String fileName. CaesarIntercept should produce a file xxxCracked.txt upon the command line call

java CaesarIntercept xxxEncoded.txt

where xxxEncoded.txt is the file that was intercepted. For this class, you are allowed to display a reasonable amount of text to the user through the console, as well as ask for feed-back on the console. If your program requires user input, make sure to include instructions (either printed to the console or given in a readme file). You must display no more than fifteen lines to the user at once, and you cannot ask the user for feedback more than five times. Finally, you may not ask the user for the key!

As with the traditional decoders, your CaesarIntercept should produce a decoded file of the same name as the read in encoded file, with the substring 'Cracked.txt' attached to the file name less 'Encoded.txt'. The new text file should appear in the folder containing xxxEncoded.txt and your Java files.

Specific Requirements

- Must take encoded file's filename as the argument to main method
- Produced text file must be xxxCracked.txt where xxxEncoded.txt is the name of the file passed in.
- Display instructions in the console or include a readme file in your submission if you require user feedback
- Can ask user for feedback from the console no more then 5 times
- Do not display more than 15 lines of text to the user at once
- Cannot ask user for the key

What to turn in
Turn in all the files you used to write this assignment into t-square. For this assignment, please submit:

- CaesarEncode.java
- CaesarDecode.java
- CaesarIntercept.java
- README.txt (only if your CaesarIntercept requires user feedback and it does not print the instructions to the console)

By submitting this assignment, you confirm that you understand the CS1332 collaboration policy, and that your submission complies with it.

Keep in mind that you can submit the assignment as many times as you want before it is due. If you find yourself working close to the time the assignment is due then please submit often since t-square has a habit of going down at the worst possible times.

Remember
These are required, and you will lose points if you do not do any of the following:

- Collaboration statement at the top of EVERY file you create.
- Class header Javadoc for EVERY file you create.
  - @author tag is required and must have your first and last name
- Javadoc for EVERY method (including getters and setters). Your javadoc should immediately inform readers what your method does.
  - @param tags are required where applicable
  - @return tag is required where applicable