Introduction to Java Programming

Day 2: Homework

Recap

Before starting your homework, take some time to review the following things we covered in class today:

- Writing and using functions, including the use of helper functions, local and global variables
- Declaring and manipulating one dimensional arrays
- More `String` and `char` manipulation, including ASCII encoding
- Using a `for` loop, including `continue`, particularly in the context of solving various algorithmic problems
- Using the Java `Math` library

Installing JDK and Eclipse

As you know, we're starting our Mini Project this Wednesday. To do this Mini Project, you will need to have a JDK (Java Development Kit) and an IDE (Interactive Development Environment) such as Eclipse installed on your personal computer. Follow the instructions in the link below to install both a JDK and Eclipse on your computer:
http://www.ntu.edu.sg/home/ehchua/programming/howto/eclipsejava_howto.html

Homework

In class, we got started on building a mini cryptography program in Java using an online Java IDE/compiler (link: https://repl.it/languages/java). For homework, you'll add the following features to this program:

1. We'll now add a new type of cipher called the Atbash cipher. Essentially, the Atbash cipher flips the order of the letters in the alphabet, so "A" becomes a "Z", a "B" becomes a "Y", and so on.
   a) Write a function with the signature `public static String encryptAtbash(String message)` that performs the encryption process described above.
   b) Write a function with the signature `public static String decryptAtbash(String message)` that decrypts a message which has been encrypted with the Atbash cipher. *Hint: use part (a)!*
2. We'll now add another type of cipher called the Route cipher. Instead of replacing each letter of the message by another, Route shuffles them in a predetermined fashion. The Route cipher works as follows:
   - Start at the first letter of the unencrypted message and move forward *k* letters at a time (*k* is some constant).
   - Once you reach the end of the string (ie. you cannot advance *k* letters without going past the end of the message), start back at the beginning of message, but now at the **second** letter.
   - Once you reach the end of the string, start back at the beginning of message, but now at the **third** letter.
   - … (continue in a similar fashion).
   - Once you've iterated over every letter of the original message, rewrite your original message's letters in the order that you iterated the letters over. The result is your encrypted message.

   For example, if our original message was PROGRAM and *k* = 4, then our encrypted message is PRRAOMG.
   a) Write a function with the signature `public static String encryptRoute(String message, int k)` that performs the encryption process described above.
   b) Write a function with the signature `public static String decryptRoute(String message, int k)` that decrypts a message which has been encrypted with the Route cipher.
3. Test and debug the new functions you wrote in a similar way to how you tested the Caesar cipher in class.