ASSIGNMENT 2
Cryptography

COMP-202, Summer 2011

Due: Monday, May 30, 2011 (23:30)

You must do this assignment individually and, unless otherwise specified, you should follow all the general instructions and regulations for assignments. Graders have the discretion to deduct up to 10% of the value of this assignment for deviations from the general instructions and regulations.

Before starting this assignment, you should download the files web2 and `SentenceChecker.java` from the course website. These files should be placed in the same folder as the file you will create called `Cryptography.java`

Part 1:	0 points
Part 2, Question 1:	20 points
Part 2, Question 2:	10 points
Part 2, Question 3:	30 points
Part 2, Question 4:	20 points
Part 2, Question 5:	20 points
	100 points total

Part 1

The questions in this part of the assignment will be graded. All of your code in this assignment should go into a class `Cryptography`.

Beginning in this assignment, the 75% non-compilation penalty will apply. This means that if your code does not compile, you will receive a maximum of 25% for the question. If you are having trouble getting your code to compile, please contact Dan or one of the TAs or consult each other or the discussion boards.

Question 1: Encrypting a String with a Caesar shift (20 points)

A simple way to encrypt things is by shifting all the letters by a given amount n. All characters other than letters (e.g. numbers, punctuation, etc. remain the same). For example, when employing a Caesar shift of 3 on the String `go rangers` you would get `jr udqjhuv`

Write a method `caesarEncrypt` which takes as input a `String originalMessage` and an `int shift` and returns the `String` created by applying a shift of `shift` to the original message. Note that your method should work with all positive and negative integers. In the case that shift is greater than 26, you should cycle over the alphabet an additional time. For example the letter a shifted by 30 is the same as the letter a shifted by 4, which would be e

Two useful methods for you will be the method `charAt()` and `length()` These methods are each defined on String. The way they work is you write the name of the variable with the String, then a dot, and then the method followed by arguments. For example, if you have:

`String s = ‘‘hello"`

then

`s.charAt(x)`

will return the xth character in s. The counting starts from 0. So for example:

`System.out.println(s.charAt(0))`

will print the letter h.

`System.out.println(s.charAt(3))`

will print the letter l.

`System.out.println(s.charAt(5))`

will give you a run time error because the String s does not have a 5th character (starting from 0)

In addition, you can get the length of a String by using the method .length . To do this, for example, write

`int length = s.length()`

One final thing is that 2 add 2 characters, you may need to cast the character to an int. For example:

`char d = 'A' + (char)3`

now the variable d will store the unicode for the letter 'D'

Question 2: Decrypting a String (10 points)

Write a method `caesarDecrypt` which takes as input a `String encoded` and an `int shift` and returns the `String` created by applying a *negative* shift of `shift` to the original message.

`caesarEncrypt()` is the inverse of `caesarDecrypt()`. This means that for any `String s`, `caesarDecrypt(caesarEncrypt(s))` is the same as `s`

Question 3: Cracking the code! (30 points)

Write a method `crackCyphar()` which takes as input a `String encoded` and an `int numberLetters`. It should then call the method `caesarDecrypt()` with all different shifts from 0...$numberLetters - 1$. After getting the result from this call, it should call the method `countEnglishWords()` given in the file `SentenceChecker` . This method takes as input a String and outputs the number of English words in it. The method provided to you will take care of things such as removing punctuation. This means, for example that `he!!!!!!llo` will count as 1 English word.

Your method should return a `String` representing the decryption with the most English words. **UPDATE**In the case of a tie, you can pick whichever one you like.

To make this work correctly, you will have to make sure that you put the file `web2` as well as `SentenceChecker.java` in the same directory as your code. Both of these files can be found on the course webpage. When you compile your code, you should type `javac SentenceChecker.java Cryptography.java` to compile.

Question 4: Generating a Random Mapping (20 points)

Write a method `generatePermutation()` which takes an int `number` as input and outputs a `int[]` . The `int[]` should be created and filled according to the following rules.

1. It should have size of exactly `number`
2. It should contain the numbers 0...$number - 1$ once and only once each
3. Each number should be chosen using the Random library class. Examples of how to use this will be discussed in class an posted on the course website.

Your method should return this array.

Question 5: Encrypting using a random permutation (20 points)

Write a method `permuteEncrypt()` which takes as input a `String input` and returns a String which represents an encoded `String`. You should do this by first calling the method you wrote in the previous section `generatePermutation()` with the input of 26. Then, for every letter, you can figure out the mapping of the i^{th} letter, by figuring out the value of the i^{th} value of the array.

For example, the letter A is the 0^{th} letter in the alphabet. To figure out what letter it maps to, get the value at the 0^{th} element of the returned array and convert this back to a letter.

Note that A is considered the 0^{th} letter, B the 1^{st} letter, etc until Z, which is the 25^{th} letter.

What To Submit

`confession.txt` - You should write in this file any information that you think is useful for the TA to mark the assignment. This should include things you were not sure of as well as parts of your code that you don't think it will work. Of course, like a confession, this will draw the TA's attention to the part of your code that doesn't work, but he/she will probably be more lenient than if he/she has to spend a lot of time looking for your error. It demonstrates that even though you couldn't solve the problem, you understand roughly what is going on.
`Cryptography.java`