CSCI 301, Fall 2013

Project #2

Vigenere Ciphers

Due date: Friday, November 22.

General instructions:

- This project should be written up in a *report.*
- This should be written in English with complete sentences and typeset nicely.
- You can use Word or Latex or any word processor you are comfortable with.
- It should document all assumptions you make, the algorithms you use, the data structures you use, and any justifications for using these.
- Any time you used output from the program to make a decision or a guess, include that output in your discussion.
- Your source code, which should be well documented with comments, should be included as an appendix. Cite procedures here in your discussion as appropriate.
- Source code should be formatted to fit well within 80 columns. Do not typeset it in landscape mode or shrink the font to overcome this limitation. Wide columns make for difficult reading. No lines should wrap to the next line when printing.

Caesar ciphers: First, write a function to decipher Caesar ciphers using frequency counts.

Caesar ciphers are a simple way of enciphering messages, and the basis of the secret decoder ring. A shift of the alphabet is used. For example, a shift of 3 letters would encipher as follows:

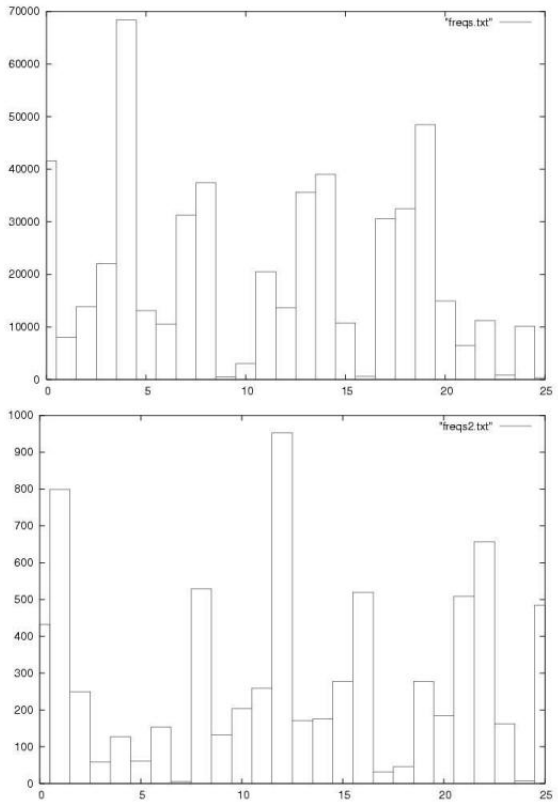
Plaintext:	A	B	C	...	W	X	Y	Z
Ciphertext:	D	E	F	...	Z	A	B	C

With a shift of 3, "HELLO WORLD" becomes "KHOOR ZRUOG". (Klingon, I think.)

Since there are only 26 Caesar ciphers, it would be easy to just try them all and look for the right one, but we need to be a little more sophisticated than that (for the next part).

We will instead rely on *frequency analysis.* Given a known frequency of letters, the frequency of letters in a ciphertext (of sufficient length) should "match up" with the known frequencies.

For example, here is the frequency of letters from a long sample of a given author's writing, and, below it, the frequency of letters from an enciphered sample of the same author's writing:



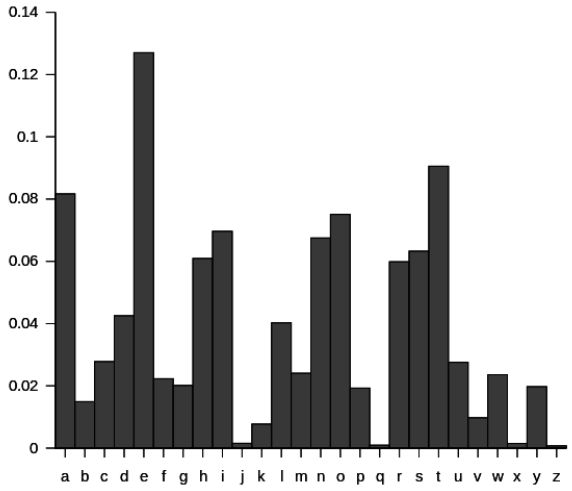
The histograms are not exactly the same, but we can see similarities. The tallest in each is probably "E", which would indicate a shift from position 4 to position 12, a shift of 8. This is strengthened by the low "trough" UVWXYZ followed by the relatively high "A" which seems to occur starting at position 2.

By examining these correlations between the histograms, you can come up with a good guess for the shift in the Caesar cipher.

In the file `enciphered.txt` I have provided a (large) enciphered text enciphered with a Caesar cipher. Usually only uppercase letters are used in this kind of cryptography, so I have left everything uppercase. Usually all puncutation is removed, as well, but I have left it in so the output will be more clear when you finally get it.

Your first assignment is write a scheme program to count the frequency of letters in the above

enciphered work (skipping any non-alphabetic characters), and compare it to a standard frequency chart, such as this one from Wikipedia:



A simple file-copy program is in `filecopy.rkt`, which should show you everything you need to know about reading single characters from a file. If you prefer, you can convert the entire file to a string and work with that:

```
(require racket/port)
(define (file->string fname)
  (port->string (open-input-file fname)))
```

The program should print out histograms of relative frequency for the alphabet (they don't have to be fancy ones like above, text output with lines of asterisks will suffice). Once you get a histogram, take a careful look at it and try to guess what shift value I used for my ciphertext. Decipher the above ciphertext using your best guess for the shift value of the Caesar cipher to see if you guessed right. (Remember to translate an enciphering shift to a deciphering shift. If a shift of n is used to encipher a text, what do you shift by to decipher?)

I have given you a huge ciphertext, the shift in the histogram should be obvious. If you want a better challenge, try just the first 1000 characters or so of the ciphertext.

If you guessed wrong and you don't see English after deciphering, look at the histogram and try another one, but RECORD all your tries and explanations. Don't just guess, look at the histogram and make intelligent guesses. This will be important later on.

Write up a brief account of your guesses and the results. Who wrote the enciphered text?

Vigenere ciphers:

Now that Caesar ciphers are done (they should have been easy), and you have some procedures to take a string and give you a histogram of frequencies so you can guess the enciphering shift, you are ready to write some functions to help you decipher Vigenere square cyphers.

Vigenere squares were once thought to be an unbreakable code. Charles Babbage figured out how to break them.

Here's how they work. Let us suppose that we have a procedure to encode with Caesar cyphers. For example, `encode(C, F)` will encode the character `C` with the Caesar cypher that begins with `F`:

A	B	C	D	E	F	G	...
F	G	H	I	J	K	L	...

and so `encode(C, F)` ⇒ `H`, `encode(F, F)` ⇒ `K`, `encode(D, F)` ⇒ `I`, *etc.*

Now, we can encode a long text and break up the frequencies by using a different Caesar cypher for each letter in the text. The letters of a *keyword* (repeated as necessary) tell us which Caesar cypher to use for each letter of the input. For example, to encode the text `HELLO WORLD` using the key `CAT`, we use the following table:

Plaintext:	H	E	L	L	O	W	O	R	L	D
Keyword:	C	A	T	C	A	T	C	A	T	C
Ciphertext:	J	E	E	N	O	P	Q	R	E	F

Thus, we encode `H` using the `C` cypher, getting `J` `E` using the `A` cypher, getting `E` *etc.* So, `HELLOWORLD` Vigenere enciphered with `CAT` is `JEENOPQREF`. For another example, the phrase `I LOVE SIR JUSTIN` encoded with the keyword `SHREK` (you can tell how long ago I made up this exercise) becomes `A SFZO KPI NEKAZR` using the pairing

Plaintext:	I	L	O	V	E	S	I	R	J	U	S	T	I	N
Keyword:	S	H	R	E	K	S	H	R	E	K	S	H	R	E
Ciphertext:	A	S	F	Z	O	K	P	I	N	E	K	A	Z	R

The great thing about Vigenere cyphers is that the frequency of letters is broken up. Since an `E`, for example, is encoded by several different letters, no one of them will dominate the frequencies, and so a Vigenere cypher cannot be broken by simple frequency analysis.

I have provided a large Vigenere-enciphered text in `vigenere.txt`. Use your previous work to print the frequency chart for this document. Can you decipher it using frequency analysis? This is why the cipher was thought to be unbreakable.

A **Vigenere Square** is a table like the one at the bottom of this document. It makes it easy to compute by hand `encode(G, K)`: you just use the `G` column and the `K` row and get `Q`. It was important that a code could be easily memorized (just remember the keyword), and easily computed by hand (by a spy hiding in a basement without access to equipment).

Breaking Vigenere ciphers:

What Charles Babbage realized, however, was that a Vigenere cypher was really just n Caesar cyphers, where n is the length of the keyword. If we can guess the length of the keyword somehow, we can use frequency analysis on every nth letter of the cyphertext!

How do we guess the length of the keyword? Consider the keyword `CAT`, and a rather long message, such as `MEET ME AT THE BOAT AT THE WHARF` The cyphertext is: `OEXV MX CT MJE UQAM CT MJE PJAKH`.

The thing to notice here is that the cyphertext has a repeated string in it: `CT MJE`. This is because the keyword is repeated, and the phrase `AT THE` is also repeated, and they line up:

M	E	E	T	M	E	A	T	T	H	E	B	O	A	T	A	T	T	H	E	W	H	A	R	F
C	A	T	C	A	T	C	A	T	C	A	T	C	A	T	C	A	T	C	A	T	C	A	T	C
O	E	X	V	M	X	C	T	M	J	E	U	Q	A	M	C	T	M	J	E	P	J	A	K	H

Certain phrases in English are very common. Since the Vigenere cypher is designed to repeat the keyword, it is only probable that in a long message some English phrase and the keyword will match up more than once. In this case, they matched up 9 characters apart, because we had 3 copies of `CAT` between the starts of the two phrases. In fact, *every* such repeated sequence must be some multiple of the keyword length. In other words, if the keyword length is n, and the distance between two repeated phrases is k, then $k \bmod n = 0$. (Why does this remind me of the pumping lemma?)

Suppose we find a cyphertext and we find two substrings repeated at a distance of 25 characters, another two repeated at a distance of 30 characters, and another two repeated at a distance of 15 characters. What's a good guess for the length of the keyword? Five, of course, because all of these numbers are zero (mod 5).

So, write a program to check for repeated substrings in a text and return the differences in their starting positions. (It will be very helpful at this point to use a version of the text that has had all non-alphabetic characters removed.) Go through the text finding all substrings of length 5, and store their position in the document in a hash table. Have the program print out the *differences* between each pair of identical string positions. The text is very long; you won't have to go through the whole thing to find lots of repetitions.

Print them out so you can examine them. If the keyword is not very long the common divisor should be obvious by inspection. If you need some help you can divide all of them by prime numbers from 3 to 19, and make a note when all (or most) are zero.

Based on this table, make a guess, n, as to the length of the keyword. Then generate n frequency tables for the "mod n" equivalence classes of the ciphertext. (Every nth character starting with the first, every nth character starting with the second, *etc.*) For each one, display a letter frequency histogram. Make an intelligent guess as to the shift value for that character in the keyword. This should give you all the letters of the keyword. (It also helps to know the keyword is an English word.)

You should now know the keyword used to encipher my text. You need to flip this keyword so that it deciphers instead of enciphers. Now you can decipher my text (at this point you might want to go back to the copy that still has the non-alphabetic characters in it).

Did you decipher it correctly? If not, there may be a mistake in your programming, or you might have guessed wrong for some of the letters in the keyword. Keep trying.

Who wrote the enciphered text? What is it?

A Vigenere Square:

```
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
B C D E F G H I J K L M N O P Q R S T U V W X Y Z A
C D E F G H I J K L M N O P Q R S T U V W X Y Z A B
D E F G H I J K L M N O P Q R S T U V W X Y Z A B C
E F G H I J K L M N O P Q R S T U V W X Y Z A B C D
F G H I J K L M N O P Q R S T U V W X Y Z A B C D E
G H I J K L M N O P Q R S T U V W X Y Z A B C D E F
H I J K L M N O P Q R S T U V W X Y Z A B C D E F G
I J K L M N O P Q R S T U V W X Y Z A B C D E F G H
J K L M N O P Q R S T U V W X Y Z A B C D E F G H I
K L M N O P Q R S T U V W X Y Z A B C D E F G H I J
L M N O P Q R S T U V W X Y Z A B C D E F G H I J K
M N O P Q R S T U V W X Y Z A B C D E F G H I J K L
N O P Q R S T U V W X Y Z A B C D E F G H I J K L M
O P Q R S T U V W X Y Z A B C D E F G H I J K L M N
P Q R S T U V W X Y Z A B C D E F G H I J K L M N O
Q R S T U V W X Y Z A B C D E F G H I J K L M N O P
R S T U V W X Y Z A B C D E F G H I J K L M N O P Q
S T U V W X Y Z A B C D E F G H I J K L M N O P Q R
T U V W X Y Z A B C D E F G H I J K L M N O P Q R S
U V W X Y Z A B C D E F G H I J K L M N O P Q R S T
V W X Y Z A B C D E F G H I J K L M N O P Q R S T U
W X Y Z A B C D E F G H I J K L M N O P Q R S T U V
X Y Z A B C D E F G H I J K L M N O P Q R S T U V W
Y Z A B C D E F G H I J K L M N O P Q R S T U V W X
Z A B C D E F G H I J K L M N O P Q R S T U V W X Y
```