实验一 Cracking Vigenère Cipher

课程名称：《信息安全》SOFT130018.01

任课老师：李景涛

助教：雷哲

实验目的

1. Understanding Vigenère Cipher
2. Understanding Kasiski Test
3. Understanding Friedman's Index of Coincidence Test
4. Understanding Frequency Analysis to crack Caesar Cipher.

实验内容

> *Note: the alphabet through the entire experiment is the uppercase English A-Z , no space, no punctuations.*

In this experiment, we will examine a well-known classical cipher, Vigenère Cipher. Dating back to 16th century , Vigenère Cipher is a polyalphabetic substitution cipher.

We will use some techniques and write computer program to cryptanalyze (to Crack!) the Vigenère Cipher.

An overview of the process is as follows. Inputting ciphertext encrypted by Vigenère Cipher and using Kasiski examination + Friedman test to determine the keyword length. Then using the ciphertext separated by the length of the keyword as a group, and crack it by analyzing the Caesar cipher. Finally, outputting the keyword and the corresponding plaintext.

Vigenère Cipher

Suppose we have an -character alphabet $A = (a_1, a_2, \dots, a_n)$, an m-character key,$K = (k_1, k_2, \dots, k_m)$ and a t-characterplaintext, $M = (m_1, m_2, \dots, m_t)$.

The alphabet are encoded by integer number, i.e. the letters A – Z are taken to be the numbers 0–25 .

Then, we define a **Vigenère Cipher** :

$$E_k(a_i) = m_i + k_i \ (mod\ n)$$

and

$$D_k(c_i) = c_i - k_i \ (mod\ n)$$

	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
A	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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

Above is the polyalphabetic substitution table of Vigenère Cipher, called the Vigenère square or Vigenère table, also known as the tabula recta.

With a random key equal to the size of the plaintext, Vigenère Cipher becomes a *one-time pad* which is the only proven unbreakable cipher in the history. If the keyword has only one character, the Vigenère Cipher degenerates into a *Caesar Cipher*.

The primary weakness of the Vigenère cipher is the repeating nature of its key. If a cryptanalyst correctly guesses the key's length, the cipher text can be treated as interwoven Caesar ciphers, which can easily be broken individually. The Kasiski examination and Friedman test can help to determine the key length.

Kasiski Test

Kasiski test is based on the observation that if two identical fragments of length three or more (trigraphs in this experiment) appears in the ciphertext, it is likely that they correspond to the same plaintext fragment. The distance between the occurrences has to be a multiple of the period.

Then we can find the common factors of the value of distance from previous occurrence, and use them as the key length candidates.

Key: *ABCDABCDABCDABCDABCDABCDABCD*

Plaintext: ***CRYPTO****ISSHORTFOR****CRYPTO****GRAPHY*

Ciphertext: ***CSASTP****KVSIQUTGQU****CSASTP****IUAQJB*

Index of Coincidence(IC,重合指数) & Friedman Test

Let $x = \{x_1, x_2, \dots x_n\}$ be a string of n alphabetic characters. The **index of coincidence** of x, $IC(x)$, is the probability that randomly chosen two elements of string x are the same.

$$IC = \frac{\sum_{i=1}^{c} n_i(n_i - 1)}{N(N-1)}$$

where N is the length of the text and n_i is the frequency (as integer) of the c letters of the alphabet (c=26 for monocase English). The sum of the n_i is necessarily N.

Coincidences in ciphertext can be caused by coincidences in the underlying plaintext, this technique is used to cryptanalyze the Vigenère cipher.

The Friedman test (sometimes known as the kappa test) was invented during the 1920s by William F. Friedman, who used the index of coincidence, which measures the unevenness of the cipher letter frequencies to break the cipher.

$$L \approx \frac{k_p - k_r}{\mathrm{IC} - k_r} = \frac{0.027\mathrm{N}}{\mathrm{IC} * (\mathrm{N} - 1) - 0.0385\mathrm{N} + 0.0655}$$

L is the key length.

The probability k_p is for any two randomly chosen source language letters are the same (around 0.0655 for monocase English)

k_r is the probability of a coincidence for a uniform random selection from the alphabet (1/26 = 0.0385 for English).

This formula applies only to the ciphertext of English plaintext. For other languages, the constants may be different.

实验步骤与过程

1. Computing the chance for the false positive rate of Kasiski test

Read the paper ***Kasiski's Test: Couldn't the Repetitions be by Accident?***

and answer the question below:

(The purpose of solving this probability is to recognize the validity of the Kasiski test.)

The likelihood of two same trigraphs (three-letter sequences) **not** being from the same plaintext fragment? n=26

Then implementing the functions below in the program `crack_vigenere_scaffold.py` to finish the experiment.

2. Implementing Vigenère Cipher

```
def vigenere_encrypt(plaintext, key, alphabet):
    """Returns ciphertext encrypted by vigenere cipher using key"""

def vigenere_decrypt(ciphertext, key, alphabet):
    """Returns plaintext decrypted by vigenere cipher using key"""
```

3. Determining the Key Size

```
def kasiski_test(ciphertext):
    """Finds gcd of most common distances between repeated trigraphs
    Recommended strategy: loop through the ciphertext, keeping a list
of trigraphs and a list of distances in this way:
    1) When encountering a new trigraph add it to the trigraph list
    2) When encountering a repeat add the distance from current index
to first index of that trigraph to the list of distances"""
```

```
def index_of_coincidence(ciphertext, alpha):
    """Caculates index of coincidnece of ciphertext"""
```

4. Determining the Keyword

Once the length of the key is known, the ciphertext can be rewritten into that many columns, with each column corresponding to a single letter of the key. Each column consists of plaintext that has been encrypted by a single Caesar cipher. The Caesar key (shift) is just the letter of the Vigenère key that was used for that column. Using methods similar to those used to break the Caesar cipher, the letters in the ciphertext can be discovered.

```
def find_likely_letters(coset, alpha, eng_freq):
    """Finds the most likely shifts for each coset and prints them
    Recommended strategy: make a list of the frequencies of each
letter in the coset, in order, A to Z.
```

```
    Then, alternate using the find total difference method (on your
frequencies list and the standard english
    frequencies list) and the rotate list method to fill out a new
list of differences.  This makes a list of
    the total difference for each possible encryption letter, A to Z,
in order.
    Then, find the indices of the smallest values in the new list,
and i2c them for the most likely letters."""
```

Hint: The shift which leads to total difference closest to the English frequency distribution is the most likely Caesar cipher key.

实验要求和评分

- 学号尾号为奇数的同学分析文本 odd，学号尾号为偶数的同学分析文本 even

```
odd =
"DWFOSZPFWGVVMVOBXTBMGIILECURUWKGZGNMUHUHMKDTNUWDRMGHPTQABVVKOTVOAMK
HCGJWGUGMEWQGIOAGSCLVSLGIQKFDIEUQXUHWCSWVGGARMMVVFWAJKICMUROBLYVQYFB
AGGGFUMFYCZXTEXNZMAPCZJTWENWLVHZNOATEHQBOABVGBVMTLWTNRSAYTCUGIMBPVME
FVMYSIXOMLUSABGBAGHZHTBUCGMQNWWGZKBNXEGHMYZVHPFMIFZLKPTRUZTPGIPUQHPG
IEFVHVFMNMTRRCAFJJEGGQADMYKBIADQTNWVFUQMWHQBOAVCBVBUIOQWLZFLBCHQAHLB
UDCGFAMJSKBTBHHAMQJIMKCVVOKKGOARTBKCBANDBBQBKBTBLNWUVUQGIHPRNQGKACZQ
ZTEHQPBTMTOVFBKMKCVFJHXCBLPVBMKBOBGNMJSXBTABDWTVUGYQFAZBTEEOAHBTMTOV
FBKMKCVFBVWVVMEFQLCPZBBLXTQWFUQGVVMYPALQTIOJTBVMBBNIDGBWASMOGFAVCTXR
OGZFVMUTWEOWGTSDRSABDZMFFZOKQMFXQMJHPRQWLUWJVMQMACNEFDXTGIYUPXPSMQGW
KVFCFUAITSIQTUXTQPNOBLOIAGCMPCFGBGBAGWZPVAMQAMETPTUGTVOOMJSUSPZFQFMV
ONHTAIGJWGVVIAUPXAKWHMLHVVMEXQLGBMREIVGFBNJVIGFKROBTISWSGZTWRQFBKVGD
BREILWBIIPQWCPTRUPXUSKBTBLCBLCBGFGBBHOKXTHIVOBBGGKNOJXCJWVEMWKBXRSAH
PPGHTQGIDPLTQVCZKHSZXPQGOVBGQAMPIIGKGURYQLVGBBNIDGDILNMGVGWIFZTECUZV
VBEOBVPVLEVIAOMEYWBUPCMCHZHTBXFDIEUG"
even =
"ZMJXCGLHBGIPSPMPSOUQEOQYIFRYWCYVBWBVQKICRMWUEDXYCVAMRRYMCSRFMBGRNTL
GTXRVJQKENAIFQZIIIJTATEHMUMVYZCZWPVHGLQUAOQDMVTBWBQAFJOXKVNQQTENVWAC
XBZRITVROUWLKJMBVRFWJXQYPWTKOIEFXKFLSBFKTTXVVFLVZKVMGQREEXQPPSEHBYIA
VIBOHCJIFBVGHFPTSOFMFPLICERITWPDBZSPLIGEHUFXGVIGUGQJTGGASEVIEHEHUDHW
MIXGKUWADTJMPMCFAVCTLCIXZVFIKMQGAQEHIKICGMSQIWIRGPBMCHAFJEKGDGROIERA
QQFBAKIOLEVVFPDMPBUWMHBYIPXKSFVQKCQYASPXZVOGRLWFWZZFWMQCAFPRRPXTGNQL
JYRITMGKMVUWBDOYHVKSHTEFVWBVRUBOBNWCIICMBVRVIDIVBUSGKMFVGMQQNOLVZGEW
DZHVKWKGQBSRZDEVBWBGKMFVATVRPRUGYVXZGPLMEGGLPCJSZFQKLMCSSZFZKWQBTSZF
ZCUTMFHKLVGVZMCWWJCUMMAFFPRRIBVUGKQJEPVQSAWIIXKGBCNVKZIPVMHUHLVZGEWD
ZHVKSHVWACXBVVEHVHERTCIFVWAZXVZGCMQCAQMKAQKSGCUWEWGLMTSRZKPGLAOAGQEI
ZIMBFLDVGQGBOPWJVXYXMBCHWGPGHZQBPXLXGKACARXGSUBBSFLLVWQYBVRZWIPFKMDY
DKZRIFWGGPIZPCGLANQGVBENZGVRVJAKMPHROMTSOFCBVFIKMQGATBUURRATXDYLKRXK
```

```
HVGGKMJIEHVHNFBJQWLBPRPIUIUXKIEHIXEKGAHORBYICOMGQUWGTKGOOAGBYIKGRSPW
QFRQYQZYHOZXKFIHRPMJWCZMGNWXIIUXVHU"
```

Hint：明文分别是两篇著名的密码学论文的第一段~

- 由于密文文本长度限制，Fieldman Test 可能不准，以 Kasiski Test 为主。
- 编程语言、编译运行等实验环境原则上不限，建议在给出的 python 框架基础上实现。如使用其他编程语言、编译运行等实验环境，需要完成同等任务并在实验报告中写明。
- 评分内容如下：

内容	**总分 100**
Probability Computation	5
Implementing Vigenère Cipher	20
Determining the Key Size(Kasiski Test + Friedman Test)	20+10
Determining the Keyword	30
Document(实验报告)	15

实验提交

- 独立完成 project，不分组
- 提交内容清单：

1. 实验报告（按**实验报告模板**书写，提交 pdf 格式文件，命名格式：学号+姓名+实验一.pdf）
2. 项目源代码（命名格式：学号_crack_vigenere.py）
3. 鼓励录制短视频，介绍代码结构、演示运行结果、分析计算量等

- 提交方式：所有提交内容以压缩文件形式上传 elearning 提交（命名格式：学号+姓名+实验一）
- 提交截止时间：3.19 日 23：59 前

参考资料：

- Wikipedia
  - Vigenère cipher
  - Kasiski examination
  - Index of coincidence