实验名称: 维吉尼亚密码加解密以及密文破解

实验目的

1. 掌握Python语言中的字符串，列表，字典，元组的使用；
2. 熟练Python语言中的条件语句和循环语句；
3. 掌握Python语言中函数的定义和使用；
4. 理解维吉尼亚密码加解密过程；
5. 理解卡西斯基试验以及如何进行频率分析。

实验内容

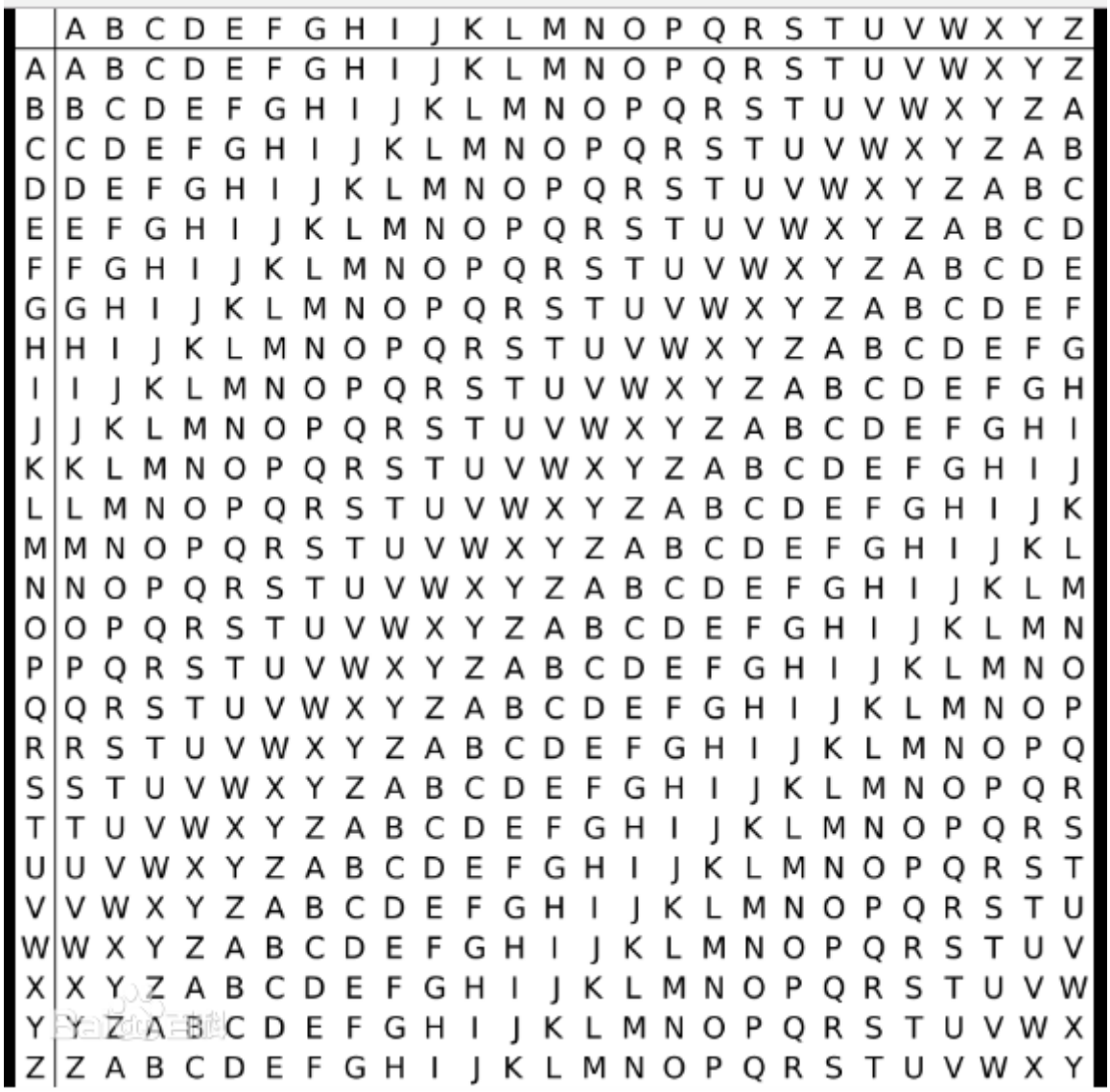
1. 编程实现维吉尼亚密码加解密过程。
2. 破解输入的维吉尼亚密文。

实验原理和步骤

维吉尼亚密码是使用一系列凯撒密码组成密码字母表的加密算法，属于多表密码的一种简单形式。

为了生成密码，需要使用表格法。这一表格包括了26行字母表，每一行都由前一行向左偏移一位得到。具体使用哪一行字母表进行编译是基于密钥进行的，在过程中会不断地变换。

表格中，第一行代表原文的字母，下面每一横行代表原文分别由哪些字母代替（也就是说每一行代表一套凯撒密码加密方法），每一竖列代表我们要用第几套字符来替换原文。一共26个字母，26套代替法，所以这个表是一个26*26的表。



	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

第一部分：维吉尼亚密码加解密

明文："Common sense is not so common."

密钥："PIZZA"

密文："Rwlloc admst qr moi an bobunm."

说明：表格中的字母都是大写，在输入输出时可以进行大小写转换，使得加解密前后明文和密文相同位置的字母大小写一致。

C (2)	P (15)	→	R (17)
O (14)	I (8)	→	W (22)
M (12)	Z (25)	→	L (11)
M (12)	Z (25)	→	L (11)
O (14)	A (0)	→	O (14)
N (13)	P (15)	→	C (2)
S (18)	I (8)	→	A (0)
E (4)	Z (25)	→	D (3)
N (13)	Z (25)	→	M (12)
S (18)	A (0)	→	S (18)
E (4)	P (15)	→	T (19)
I (8)	I (8)	→	Q (16)
S (18)	Z (25)	→	R (17)
N (13)	Z (25)	→	M (12)
O (14)	A (0)	→	O (14)
T (19)	P (15)	→	I (8)
S (18)	I (8)	→	A (0)
O (14)	Z (25)	→	N (13)
C (2)	Z (25)	→	B (1)
O (14)	A (0)	→	O (14)
M (12)	P (15)	→	B (1)
M (12)	I (8)	→	U (20)
O (14)	Z (25)	→	N (13)
N (13)	Z (25)	→	M (12)

如果用数字0-25代替字母A-Z，维吉尼亚密码的加密文法可以写成同余的形式：

加密：Ci = （ Pi + Ki) mod 26

解密：Pi = （ Ci - Ki ）mod 26

其中Pi 是原文字母下标，Ci是密文字母下标，Ki是密钥字母下标。

任务1：编码实现维吉尼亚密码加解密过程

```
# 维吉尼亚密码加解密过程
import string
from itertools import cycle

# 代码写在这里
```

```
# 测试
message = "Common sense is not so common."
key = "PIZZA"

cipher_text =
print("加密前的文本是：", message)
print("加密后的文本是：", cipher_text)

decrypted_text =
print("再次解密后的文本是：",decrypted_text)
```

```
加密前的文本是： Common sense is not so common.
加密后的文本是： Rwlloc admst qr moi an bobunm.
再次解密后的文本是： Common sense is not so common.
```

第二部分：维吉尼亚密文破解

明文中相同字符加密后可能为不同字符；真实情况下，只有密文，没有密钥，怎么破解维吉尼亚密码呢？

卡西斯基试验(Kasiski Examination)

Charles Babbage被认为破解了维吉尼亚密码，但是他没有公布结果。后来他的方法被20世纪初期的数学家卡西斯基公布出来，所以该方法就叫做卡西斯基试验。

在破解维吉尼亚密文时，我们不知道密钥长度。但是，我们知道其实密钥也就一个，密钥像滑窗一样进行加密。由于英文有很多重复的单词，比如the等等，这时候如果我们在一串密文中发现两段相同的字符串，两者之间的距离很有可能就是密钥的长度。我们多发现几段相同的字符串，求它们之间间隔的**因数**就可以大致确定密钥长度。知道密钥长度之后，维吉尼亚密码退化成为一个移位密码，可以用破解凯撒密码的方式去破解它。

问题：怎么分析密钥的长度呢？步骤如下：

1. 从密文中找出拼写完全相同的字符串；数这些相同的字符串中间间隔的字母数

举例：

- **密文如下**："Ppqca xqvekg ybnkmazu ybngbal jon i tszm jyim. Vrag voht vrau c tksg. Ddwuo xitlazu vavv raz c vkb qp iwpou."
- **移除非字母字符并转换为大写之后**，得到 PPQCAXQVEKGYBNKMAZUYBNGBALJONITSZMJYIMVRAGVOHTVRAUCTKSGDDWUOXITLAZUVAVVRAZCVKBQPIWPOU
- 发现**VRA**, **AZU**, and **YBN** 是重复序列
- PPQCAXQVEKGYBNKMAZUYBNGBALJONITSZMJYIM**VRA**GVOHT**VRA**UCTKSGDDWUOXITLAZUVAV**VRA**ZCVKBQPIWPOU
- PPQCAXQVEKGYBNKM**AZU**YBNGBALJONITSZMJYIMVRAGVOHTVRAUCTKSGDDWUOXITL**AZU**VAVVRAZCVKBQPIWPOU
- PPQCAXQVEKG**YBN**KMAZU**YBN**GBALJONITSZMJYIMVRAGVOHTVRAUCTKSGDDWUOXITLAZUVAVVRAZCVKBQPIWPOU
- **找到序列之间的间距**：
  - 第一个和第二个 VRA 序列之间间隔8个字母；
  - 第二个和第三个 VRA 序列之间间隔24个字母；

- 第一个和第三个 VRA 序列之间间隔32个字母；
- 第一个和第二个 AZU 序列之间间隔48个字母；
- 第一个和第二个 YBN 序列之间间隔8个字母。

```
def find_repeat_sequences_spacings(message):
    """
    参数：message，是一个字符串
    功能：遍历message，找到3到5个字母长度的重复序列
    返回值：一个字典，键是序列，值是序列间隔列表（重复序列之间间隔的字母数）"""
    # 代码如下
```

```
# 测试1
ciphertext = "Ppqca xqvekg ybnkmazu ybngbal jon i tszm jyim. Vrag voht vrau c
tksg. Ddwuo xitlazu vavv raz c vkb qp iwpou."
print(find_repeat_sequences_spacings(ciphertext))
# 打印结果为：{'YBN': [8], 'AZU': [48], 'VRA': [8, 32, 24]}
```

2. 找到间隔之间的因数

- 8的因数是2,4,8
- 24的因数是2,3,4,6,8,12,24
- 32的因数是2,4,8,16,32
- 48的因数是2,3,4,6,8,12,16,24,48

次数出现最多的几个因数极有可能就是密钥字符串的长度，即2,4,8,3,6,12

```
def get_useful_factors(num):
    """返回num的因子列表"""

# 测试
print(get_useful_factors(24))  # [2, 3, 4, 6, 8, 12]
```

编程计算每个因子出现的次数，按照出现次数从大到小排序，得到一个因子列表，这些因子就是可能的密钥长度

```
# 编程找到可能的密钥长度
```

3. 将密文字符串按照密钥长度分组，并获取每组中第N个字母

假设密钥长度为4，将密文按每4个字母为一组分开，提取字母：

- 从第1个字母开始0,4,8，..,从第2个字母开始1,5,9,...,从第3个字母开始2,6,10,...,从第4个字母开始3,7,11,...这样提取到4个字符串，对这4个字符串按照破解凯撒加密密文进行频率分析，计算分数。
- 如果我们从卡西斯基试验猜测是正确的，那么密钥的第一个字母对明文加密得到的就是第一个字符串；密钥的第二个字母对明文加密得到的就是第二个字符串；以此类推。
- 得到的字符串分别为：
  - 从第1个字母开始得到：PAEBABANZIAHAKDXAAAKIU
  - 从第2个字母开始得到：PXKNZNLIMMGTUSWIZVZBW
  - 从第3个字母开始得到：QQGKUGJTJVVVCGUTUVCQP

- 从第4个字母开始得到：CVYMYBOSYRORTDOLVRVPO

```
def get_nth_subkeys_letters(n,key_length,message):
    """将message按照key_length长度分组，每个分组中第n个字母拿出来
    getNthSubkeysLetters(1, 3, 'ABCABCABC') returns 'AAA'
    getNthSubkeysLetters(2, 3, 'ABCABCABC') returns 'BBB'
    getNthSubkeysLetters(3, 3, 'ABCABCABC') returns 'CCC'
    getNthSubkeysLetters(1, 5, 'ABCDEFGHI') returns 'AF
    """
```

```
ciphertext = "Ppqca xqvekg ybnkmazu ybngbal jon i tszm jyim. Vrag voht vrau c
tksg. Ddwuo xitlazu vavv raz c vkb qp iwpou."
for i in range(1,5):
    print(get_nth_subkeys_letters(i,4,ciphertext))
```

4. 对上面得到的字符串用频率分析破解得到密钥中的每个字母

假设子密钥从26个字母中选取，然后对上面的每个字符串进行解密，并计算分数，选取分数最高的几个，就是可能的子密钥。

- 第1个字符串最可能的子密钥是：A, I, N, W
- 第2个字符串最可能的子密钥是：I, Z, A, E
- 第3个字符串最可能的子密钥是：C, G, H, I
- 第4个字符串最可能的子密钥是：K, N, R, V

然后用暴力破解，对这些子密钥进行组合，共有4*4*4*4=256种可能性。

频率分析

英语字母表有26个字母，但是每个字母在英语文本中出现的频率不一样。按照出现频率从高到低排序为**ETAOINSHRDLCUMWFGYPBVKJXQZ**。

在明文和密文中计算字母总数和它们出现的频率称为**频率分析**。

当我们需要破解维吉尼亚密码时，我们不能再分析单词，因为密文中的单词可能是多个子密钥加密的，相应地我们要分析每个子密钥的加密文本的字母频率。

我们设置一个频率匹配分数score，初值为0，计算过程如下：

- 计算密文中每个字母出现的频率，从高到低排序，
- 计算前6个字母和后6个字母是否出现在ETAOIN的前6个字母处和后6个字母处，
- 出现score就加1，这样score取值是0-12。

一段密文如下：“I rc ascwuiluhnviwuetnh,osgaa ice tipeeeee slnatsfietgi tittynecenisl. e fo f fnc isltn sn o a yrs sd onisli ,l erglei trhfmwfrogotn,l stcofiit.aea wesn,lnc ee w,l elh eeehoer ros iol er snh nl oahsts ilasvih tvfeh rtira id thatnie.im ei-dlmf i thszonsisehroe, aiehcdsanahiec gv gyedsB affcahiecesd d lee onsdihsoc nin cethiTitx eRneahgin r e teom fbiotd n ntacscwevhtdhnhpiwru”

统计字母出现频率，从高到低排序如下：EISNTHAOCLRFDGWVMUYBPZXQJK。前6个字母出现了E,I,N,T, 后6个字母出现了K,J,X,Q,Z, 所以分数是9。

```
# 英语字母频率
ETAOIN = "ETAOINSHRDLCUMWFGYPBVKJXQZ"

def freq_match_score(message):
```

```
    """将message中字符按照频数从高到低排序，
    然后查看出现频数最高的前6个字符和出现频数最低的后6个字符
    是否与英语字母频数表（ETAOIN）中的字符相匹配。
    若匹配，match_score加1"""

    match_score = 0  # 初值为0，每匹配一个字符加1，取值范围[0,12]

    # 第一步：获得{字母:频数}字典
    # 形如{'A': 135, 'C': 74, 'B': 30, 'E': 196}



    # 第二步，获得{频数：字母列表}字典
    # 因为有可能有些字母出现的频数相同
    # 形如{41: ['A'], 2: ['B', 'F'], 14: ['C'], 0: ['D', 'K', 'V']}



    # 第三步，将频数相同的字母列表按照 "ETAOIN" 顺序排序，然后转化为字符串
    # 这样能保证频数相同字母输出一致


    # 第四步，将{频数：字母列表}字典转化为一个元组的列表[(freq,letters),...]
    # 然后按照元组的第一项排序



    # 第五步，字母按照频数排序好了，将第四步列表中每一个元素的第二项提取出来
    # 得到按照英语字母频率排序的字符串


    # 第六步，计算频率匹配分数并返回结果
```

```
# 测试示例中的密文，得到分数 9
message = "I rc ascwuiluhnviwuetnh,osgaa ice tipeeeee slnatsfietgi
tittynecenisl. e fo f fnc isltn sn o a yrs sd onisli ,l erglei trhfmwfrogotn,l
stcofiit.aea  wesn,lnc ee w,l eIh eeehoer ros  iol er snh nl oahsts  ilasvih
tvfeh rtira id thatnie.im ei-dlmf i  thszonsisehroe, aiehcdsanahiec  gv gyedsB
affcahiecesd d lee   onsdihsoc nin cethiTitx  eRneahgin r e teom fbiotd  n
ntacscwevhtdhnhpiwru"
print(freq_match_score(message))
```

程序化检测英语

对于置换密码，如果我们用暴力破解方法寻找明文，我们需要测试不同的密钥。只有一个密钥会输出正确的明文，其他密钥输出的是垃圾信息。一旦我们找到了明文，后面的key就不用再尝试了。

问题：电脑如何区分垃圾信息和英语文本（有实际意义的）？

电脑没法区分它们，但是我们注意到英语文本是由可以从字典查找到的单词组成的，而垃圾信息的组成单词我们找不到。如果我们有一个字典文件，每一行是一个英文单词，并且某一个密钥破解后的明文的绝大多数单词都在字典文件中，那么我们可以猜测破解后的明文大概率是我们想要的（假设明文是英语）。

```
def is_english(message, word_percentage=20, letter_percentage=85):
    """破解得到的字符串中至少要有20%的单词在字典中，
    85%的字符必须是字母或者空格（不是符号或者数字）
    word_percentage 和 letter_percentage的默认值可以调整"""

    # 第一步：加载字典文件


    # 第二步：统计message中的单词在字典中的频率
    # 假设message中有W个单词在字典文件中，message中共有 M 个单词
    # 则单词频率为 W/M


    # 假设message中字母或者空格的数量为 L，message的长度为C
    # 则字符频率为 L/C

    # 如果W/M 的值 大于或者等于 word_percentage 并且 L/C 的值 大于或者等于
letter_percentage
    # 则函数返回True，否则返回False
```

实现破解维吉尼亚密码的主程序

1. 获得密钥可能长度的列表；
2. 对于每个可能的密钥长度，暴力破解密文;
    - 获取每组第N个字母构成的字符串（N取值从1到密钥长度）；
    - 对于字母表中的每个字母，尝试用维吉尼亚解密过程解密；
    - 计算解密后字符串的频率匹配分数，确定密钥每个位置可能的字母；
    - 尝试密钥每个位置处可能字母的组合，破解密文；
    - 判断破解后得到的字符串是否为英语，输出提示信息并查看结果，通过结果判断是否继续破解。

```
# 破解密文主程序
```

用文件ciphertext_vigenere.txt（从QQ群文件下载）中的密文测试破解程序，给出明文。