002297666 Matthew Kuo

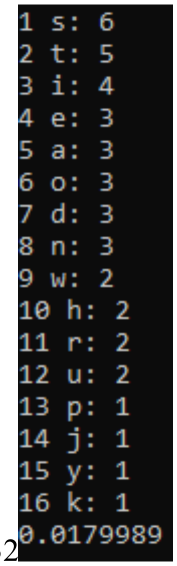
0) We've been describing alphabetic languages like English, Italian and Arabic. What happens if we look at a language like Chinese where there may be 4000 symbols? Would there be more or less information about the source on a computer representation of the language?

Since Chinese as a language also has frequency pattern similar to English (but with more different symbols), there will be more information regarding the frequency of symbols for us to analyze.

1) Calculate the phi or psi statistic on single letters and pairs of letters for ciphers 1,2,3,4 on the first handout. One should be unusual, why? (Phi is better here because the samples are small).

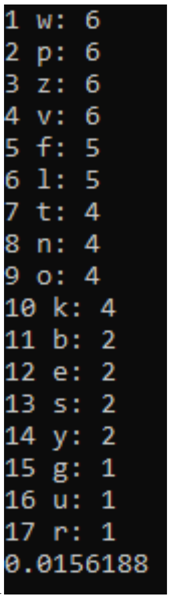
The results are calculated based on my program count_psi_phi.cpp.

Single letter:



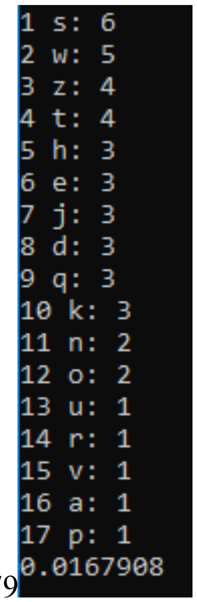
```
1 s: 6
2 t: 5
3 i: 4
4 e: 3
5 a: 3
6 o: 3
7 d: 3
8 n: 3
9 w: 2
10 h: 2
11 r: 2
12 u: 2
13 p: 1
14 j: 1
15 y: 1
16 k: 1
0.0179989
```

Phi of cipher 1: 0.0179988662132



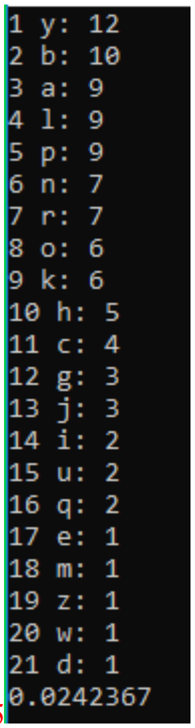
```
1 w: 6
2 p: 6
3 z: 6
4 v: 6
5 f: 5
6 l: 5
7 t: 4
8 n: 4
9 o: 4
10 k: 4
11 b: 2
12 e: 2
13 s: 2
14 y: 2
15 g: 1
16 u: 1
17 r: 1
0.0156188
```

Phi of cipher 2: 0.0156188247941



```
1 s: 6
2 w: 5
3 z: 4
4 t: 4
5 h: 3
6 e: 3
7 j: 3
8 d: 3
9 q: 3
10 k: 3
11 n: 2
12 o: 2
13 u: 1
14 r: 1
15 v: 1
16 a: 1
17 p: 1
0.0167908
```

Phi of cipher 3: 0.0167908373179

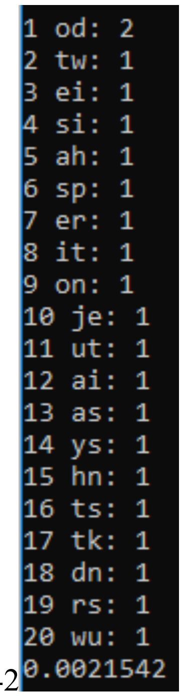


```
1 y: 12
2 b: 10
3 a: 9
4 l: 9
5 p: 9
6 n: 7
7 r: 7
8 o: 6
9 k: 6
10 h: 5
11 c: 4
12 g: 3
13 j: 3
14 i: 2
15 u: 2
16 q: 2
17 e: 1
18 m: 1
19 z: 1
20 w: 1
21 d: 1
0.0242367
```

Phi of cipher 4: 0.0242366528025

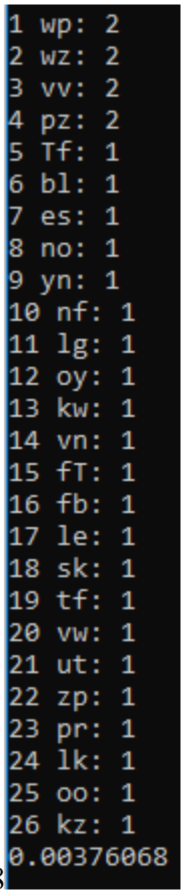
The phi value of last cipher is not normal.

Pair:



```
1 od: 2
2 tw: 1
3 ei: 1
4 si: 1
5 ah: 1
6 sp: 1
7 er: 1
8 it: 1
9 on: 1
10 je: 1
11 ut: 1
12 ai: 1
13 as: 1
14 ys: 1
15 hn: 1
16 ts: 1
17 tk: 1
18 dn: 1
19 rs: 1
20 wu: 1
0.0021542
```

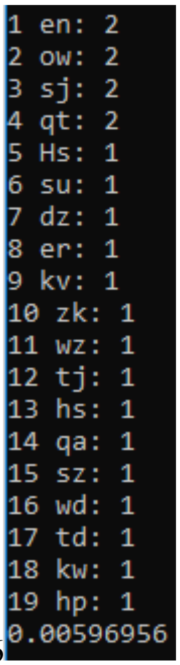
Phi of cipher 1: 0.0021542



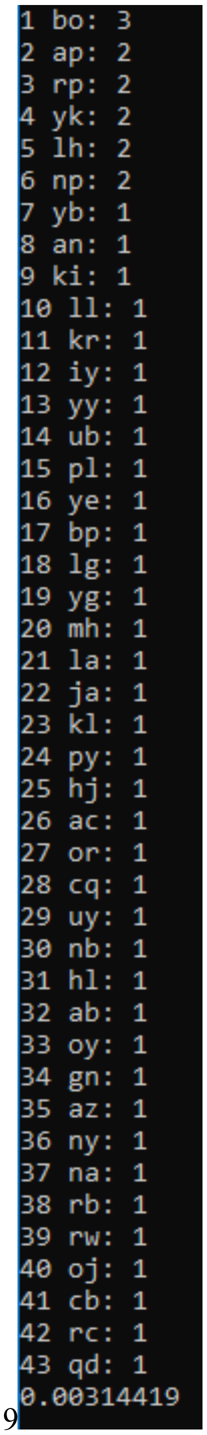
```
1 wp: 2
2 wz: 2
3 vv: 2
4 pz: 2
5 Tf: 1
6 bl: 1
7 es: 1
8 no: 1
9 yn: 1
10 nf: 1
11 lg: 1
12 oy: 1
13 kw: 1
14 vn: 1
15 fT: 1
16 fb: 1
17 le: 1
18 sk: 1
19 tf: 1
20 vw: 1
21 ut: 1
22 zp: 1
23 pr: 1
24 lk: 1
25 oo: 1
26 kz: 1
0.00376068
```

Phi of cipher 2: 0.00376068

Phi of cipher 3: 0.00596956



```
1 en: 2
2 ow: 2
3 sj: 2
4 qt: 2
5 Hs: 1
6 su: 1
7 dz: 1
8 er: 1
9 kv: 1
10 zk: 1
11 wz: 1
12 tj: 1
13 hs: 1
14 qa: 1
15 sz: 1
16 wd: 1
17 td: 1
18 kw: 1
19 hp: 1
0.00596956
```



```
1 bo: 3
2 ap: 2
3 rp: 2
4 yk: 2
5 lh: 2
6 np: 2
7 yb: 1
8 an: 1
9 ki: 1
10 ll: 1
11 kr: 1
12 iy: 1
13 yy: 1
14 ub: 1
15 pl: 1
16 ye: 1
17 bp: 1
18 lg: 1
19 yg: 1
20 mh: 1
21 la: 1
22 ja: 1
23 kl: 1
24 py: 1
25 hj: 1
26 ac: 1
27 or: 1
28 cq: 1
29 uy: 1
30 nb: 1
31 hl: 1
32 ab: 1
33 oy: 1
34 gn: 1
35 az: 1
36 ny: 1
37 na: 1
38 rb: 1
39 rw: 1
40 oj: 1
41 cb: 1
42 rc: 1
43 qd: 1
0.00314419
```

Phi of cipher 4: 0.00314419

2) (the file adfgvx.encode) is this a mono alphabetic substitution? Prove it.
It is not, if it were mono alphabetic substitution,
the psi value would be close to 0.0659251732951
and the phi value would be close to 0.0274636348336

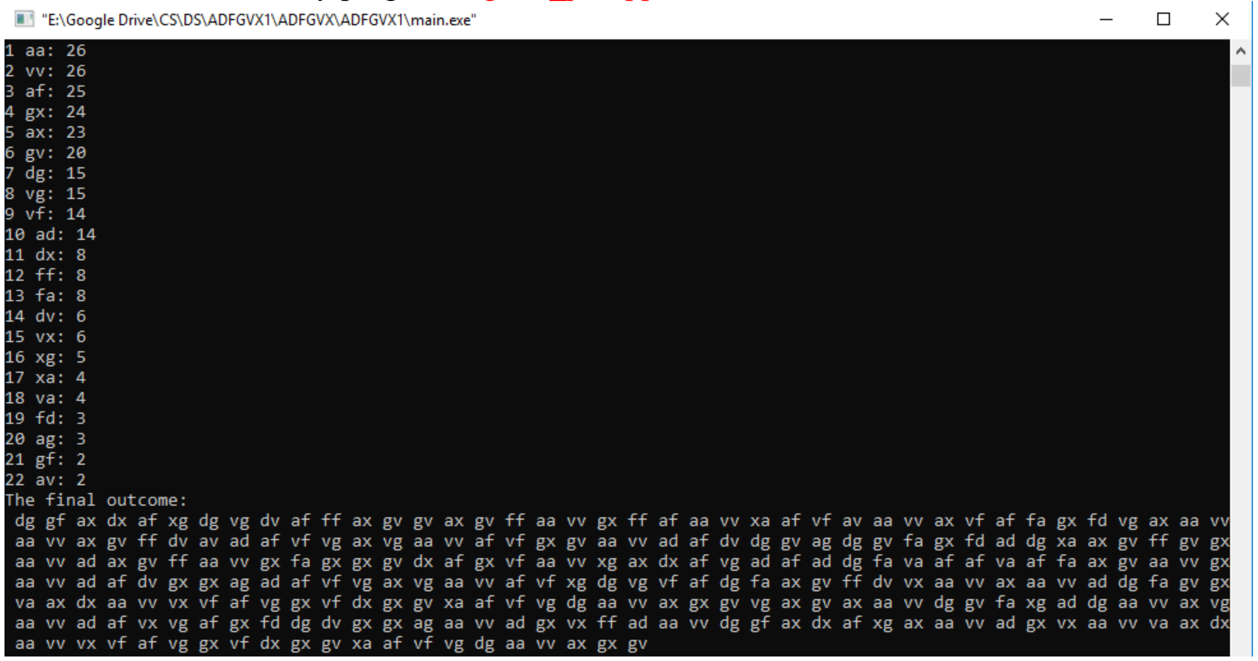
However, the psi value of the cipher is: 0.187467888023
and the phi value is: 0.0208012213561
Because the psi value is far from normal, and because the text contains only 6 letters, we can say it is not mono alphabetic substitution.

3) It could be a variant of adfgvx. Is it permuted or not? Prove it.
Yes, it is. I manage to find that when it is permuted as sequence “6321504”, I can get the highest psi value at 0.0685692.

4)Side information tells you that if it is a transposition then it is transposed in blocks of between 3 and 10 characters with any characters that are left over appended without permutation. Find the permutation (actually you'll find the inverse if you use itertools.permutations to generate all the possible permutations).
As mentioned in last answer, the permutation should be “6321504”. The text after I reverse the permutation and add a space every two letters is:

```
dg gf ax dx af xg dg vg dv af ff ax gv gv ax gv ff aa vv gx ff af aa vv xa af vf av aa vv ax vf af
fa gx fd vg ax aa vv aa vv ax gv ff dv av ad af vf vg ax vg aa vv af vf gx gv aa vv ad af dv dg
gv ag dg gv fa gx fd ad dg xa ax gv ff gv gx aa vv ad ax gv ff aa vv gx fa gx gx gv dx af gx vf
aa vv xg ax dx af vg ad af ad dg fa va af af va af fa ax gv aa vv gx aa vv ad af dv gx gx ag ad
af vf vg ax vg aa vv af vf xg dg vg vf af dg fa ax gv ff dv vx aa vv ax aa vv ad dg fa gv gx va
ax dx aa vv vx vf af vg gx vf dx gx gv xa af vf vg dg aa vv ax gx gv vg ax gv ax aa vv dg gv fa
xg ad dg aa vv ax vg aa vv ad af vx vg af gx fd dg dv gx gx ag aa vv ad gx vx ff ad aa vv dg gf
ax dx af xg ax aa vv ad gx vx aa vv va ax dx aa vv vx vf af vg gx vf dx gx gv xa af vf vg dg aa
vv ax gx gv
```

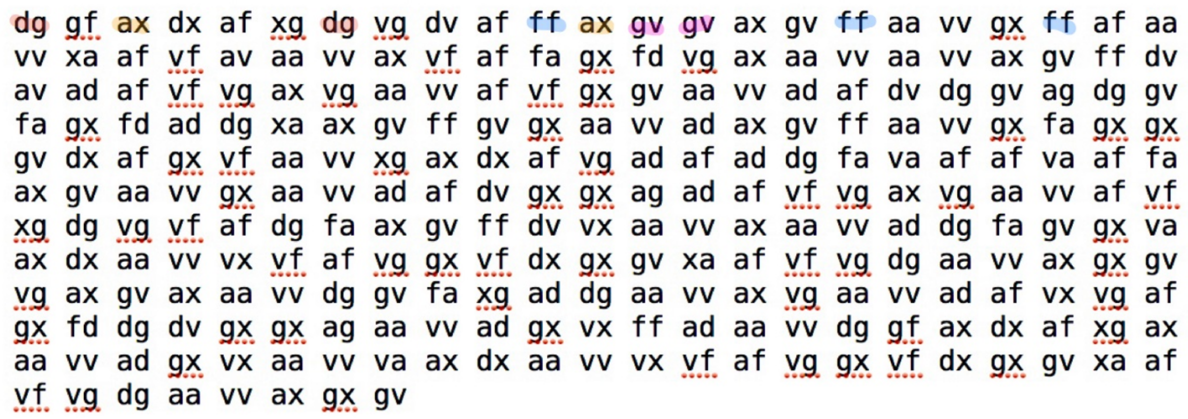
The result is based on my program highest_psi.cpp.



```
"E:\Google Drive\CS\DS\ADFGVX1\ADFGVX\ADFGVX1\main.exe"
1 aa: 26
2 vv: 26
3 af: 25
4 gx: 24
5 ax: 23
6 gv: 20
7 dg: 15
8 vg: 15
9 vf: 14
10 ad: 14
11 dx: 8
12 ff: 8
13 fa: 8
14 dv: 6
15 vx: 6
16 xg: 5
17 xa: 4
18 va: 4
19 fd: 3
20 ag: 3
21 gf: 2
22 av: 2
The final outcome:
 dg gf ax dx af xg dg vg dv af ff ax gv gv ax gv ff aa vv gx ff af aa vv xa af vf av aa vv ax vf af fa gx fd vg ax aa vv
 aa vv ax gv ff dv av ad af vf vg ax vg aa vv af vf gx gv aa vv ad af dv dg gv ag dg gv fa gx fd ad dg xa ax gv ff gv gx
 aa vv ad ax gv ff aa vv gx fa gx gx gv dx af gx vf aa vv xg ax dx af vg ad af ad dg fa va af af va af fa ax gv aa vv gx
 aa vv ad af dv gx gx ag ad af vf vg ax vg aa vv af vf xg dg vg vf af dg fa ax gv ff dv vx aa vv ax aa vv ad dg fa gv gx
 va ax dx aa vv vx vf af vg gx vf dx gx gv xa af vf vg dg aa vv ax gx gv vg ax gv ax aa vv dg gv fa xg ad dg aa vv ax vg
 aa vv ad af vx vg af gx fd dg dv gx gx ag aa vv ad gx vx ff ad aa vv dg gf ax dx af xg ax aa vv ad gx vx aa vv va ax dx
 aa vv vx vf af vg gx vf dx gx gv xa af vf vg dg aa vv ax gx gv
```

5) It may have come from Alice in wonderland. How would you check this without solving the cipher? Write a program and try it. (There are several fast algorithms that will identify too many places and a slower algorithm that will only find correct answers).

It is probably not from Alice in wonderland. My method is to check every 261 alphabets, from 1st to 261st, from 2nd to 262nd, and so on (since the cipher contains 522 characters, if it's ADFGVX, there should be only 261 characters in plaintext). And the 261 letters in Alice in wonderland should match the pattern of the cipher, which is to say, the first letter should be equal to the seventh letter, the third letter should be equal to the twelfth letter, so on and so forth.



```
dg gf ax dx af xg dg vg dv af ff ax gv gv ax gv ff aa vv gx ff af aa
vv xa af vf av aa vv ax vf af fa gx fd vg ax aa vv aa vv ax gv ff dv
av ad af vf vg ax vg aa vv af vf gx gv aa vv ad af dv dg gv ag dg gv
fa gx fd ad dg xa ax gv ff gv gx aa vv ad ax gv ff aa vv gx fa gx gx
gv dx af gx vf aa vv xg ax dx af vg ad af ad dg fa va af af va af fa
ax gv aa vv gx aa vv ad af dv gx gx ag ad af vf vg ax vg aa vv af vf
xg dg vg vf af dg fa ax gv ff dv vx aa vv ax aa vv ad dg fa gv gx va
ax dx aa vv vx vf af vg gx vf dx gx gv xa af vf vg dg aa vv ax gx gv
vg ax gv ax aa vv dg gv fa xg ad dg aa vv ax vg aa vv ad af vx vg af
gx fd dg dv gx gx ag aa vv ad gx vx ff ad aa vv dg gf ax dx af xg ax
aa vv ad gx vx aa vv va ax dx aa vv vx vf af vg gx vf dx gx gv xa af
vf vg dg aa vv ax gx gv
```

So, I used if statement to keep checking if I can find any section of text in Alice in wonderland that can match this pattern.

```
if((s[0]==s[6])&&(s[2]==s[11])&&(s[12]==s[13])&&(s[10]==s[16])&&(s[10]==s[20]))
```

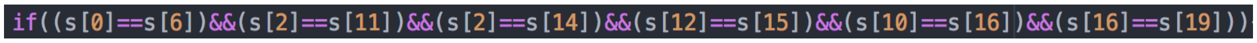
However, the program can't return a match. Therefore, I believe the cipher is not from Alice in wonderland

The method is implemented by my program find_psi_261_new.cpp.

(5 UPDATE
Professor uploaded a new ADFGVX cipher. According to the new cipher, I got the same permutation (this cipher has only 470 characters).
My answer is based on my program find_psi_235.cpp.

```
permutation: 6321504
current highest psi: 0.0723404
1 aa: 26
2 af: 25
3 gx: 24
4 ax: 23
5 gv: 20
6 dg: 15
7 vg: 15
8 vf: 14
9 ad: 14
10 dx: 8
11 ff: 8
12 fa: 8
13 dv: 6
14 vx: 6
15 xg: 5
16 xa: 4
17 va: 4
18 fd: 3
19 ag: 3
20 gf: 2
21 av: 2
The final outcome:
dg gf ax dx af xg dg vg dv af ff ax gv gv ax gv ff aa gx ff af aa xa
af vf av aa ax vf af fa gx fd vg ax aa aa ax gv ff dv av ad af vf vg
ax vg aa af vf gx gv aa ad af dv dg gv ag dg gv fa gx fd ad dg xa ax
gv ff gv gx aa ad ax gv ff aa gx fa gx gx gv dx af gx vf aa xg ax dx
af vg ad af ad dg fa va af af va af fa ax gv aa gx aa ad af dv gx gx
ag ad af vf vg ax vg aa af vf xg dg vg vf af dg fa ax gv ff dv vx aa
ax aa ad dg fa gv gx va ax dx aa vx vf af vg gx vf dx gx gv xa af vf
vg dg aa ax gx gv vg ax gv ax aa dg gv fa xg ad dg aa ax vg aa ad af
vx vg af gx fd dg dv gx gx ag aa ad gx vx ff ad aa dg gf ax dx af xg
ax aa ad gx vx aa va ax dx aa vx vf af vg gx vf dx gx gv xa af vf vg
dg aa ax gx gv
```

And use the relative positions of these pairs to find match in Alice in wonderland.



```
if((s[0]==s[6])&&(s[2]==s[11])&&(s[2]==s[14])&&(s[12]==s[15])&&(s[10]==s[16])&&(s[16]==s[19]))
```

The result that matches in Alice in wonderland is:

```
Alicewasbeginningtogetverytiredofsittingbyhersisteront
hebankandofhavingnothingtodoonceortwiceshehadpeepedint
othebookhersisterwasreadingbutithadnopicturesorconvers
ationsinitandwhatistheuseofabookthoughtalicewithoutpic
turesorconversation
```

The ADFGVX matrix should be:

	A	D	F	G	V	X
A	t	h	e	k	y	i
D				a	b	c
F	d	f	g			
G			l		n	o
V	p		r	s		u
X	v			w		