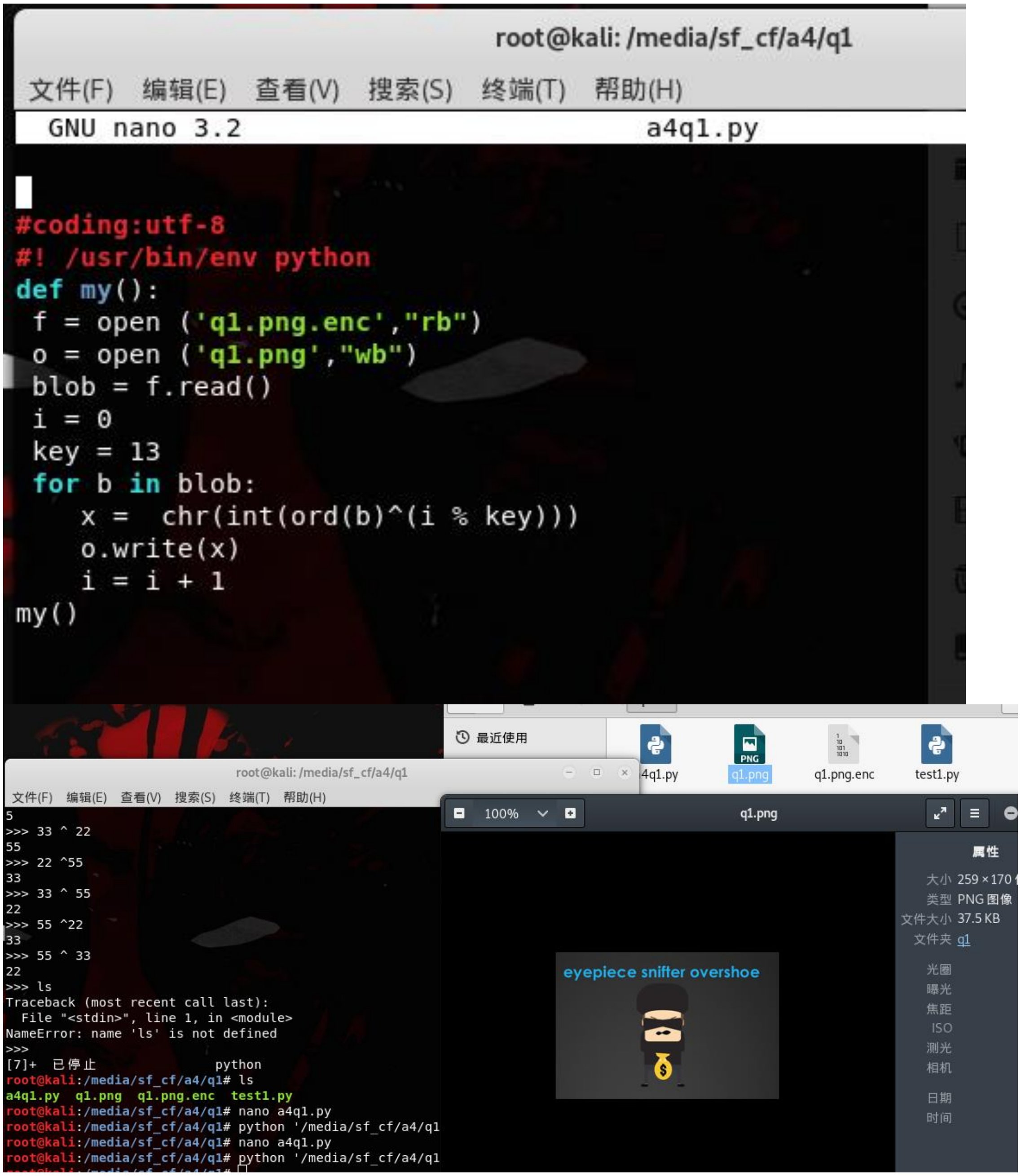
Q1.



```
#coding:utf-8
#! /usr/bin/env python
def my():
 f = open ('q1.png.enc',"rb")
 o = open ('q1.png',"wb")
 blob = f.read()
 i = 0
 key = 13
 for b in blob:
    x =  chr(int(ord(b)^(i % key)))
    o.write(x)
    i = i + 1
my()
```

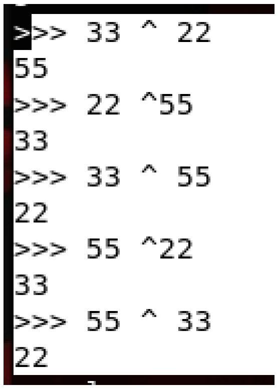
```
5
>>> 33 ^ 22
55
>>> 22 ^55
33
>>> 33 ^ 55
22
>>> 55 ^22
33
>>> 55 ^ 33
22
>>> ls
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'ls' is not defined
>>>
[7]+  已停止                 python
root@kali:/media/sf_cf/a4/q1# ls
a4q1.py  q1.png  q1.png.enc  test1.py
root@kali:/media/sf_cf/a4/q1# nano a4q1.py
root@kali:/media/sf_cf/a4/q1# python '/media/sf_cf/a4/q1
root@kali:/media/sf_cf/a4/q1# nano a4q1.py
root@kali:/media/sf_cf/a4/q1# python '/media/sf_cf/a4/q1
```

I don’t think this is a good encryption scheme, even the key was kept secret.

Reason:

a^b=c,a^c=b,c^b=a.

e.g.



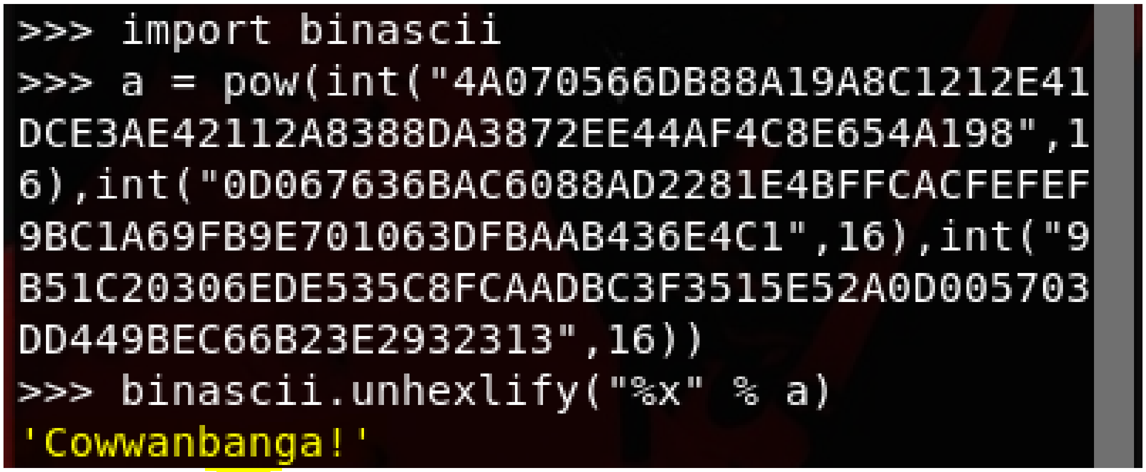
```
>>> 33 ^ 22
55
>>> 22 ^55
33
>>> 33 ^ 55
22
>>> 55 ^22
33
>>> 55 ^ 33
22
```

they can transform into each other.

Brute force will use linear time to crack the cipher.

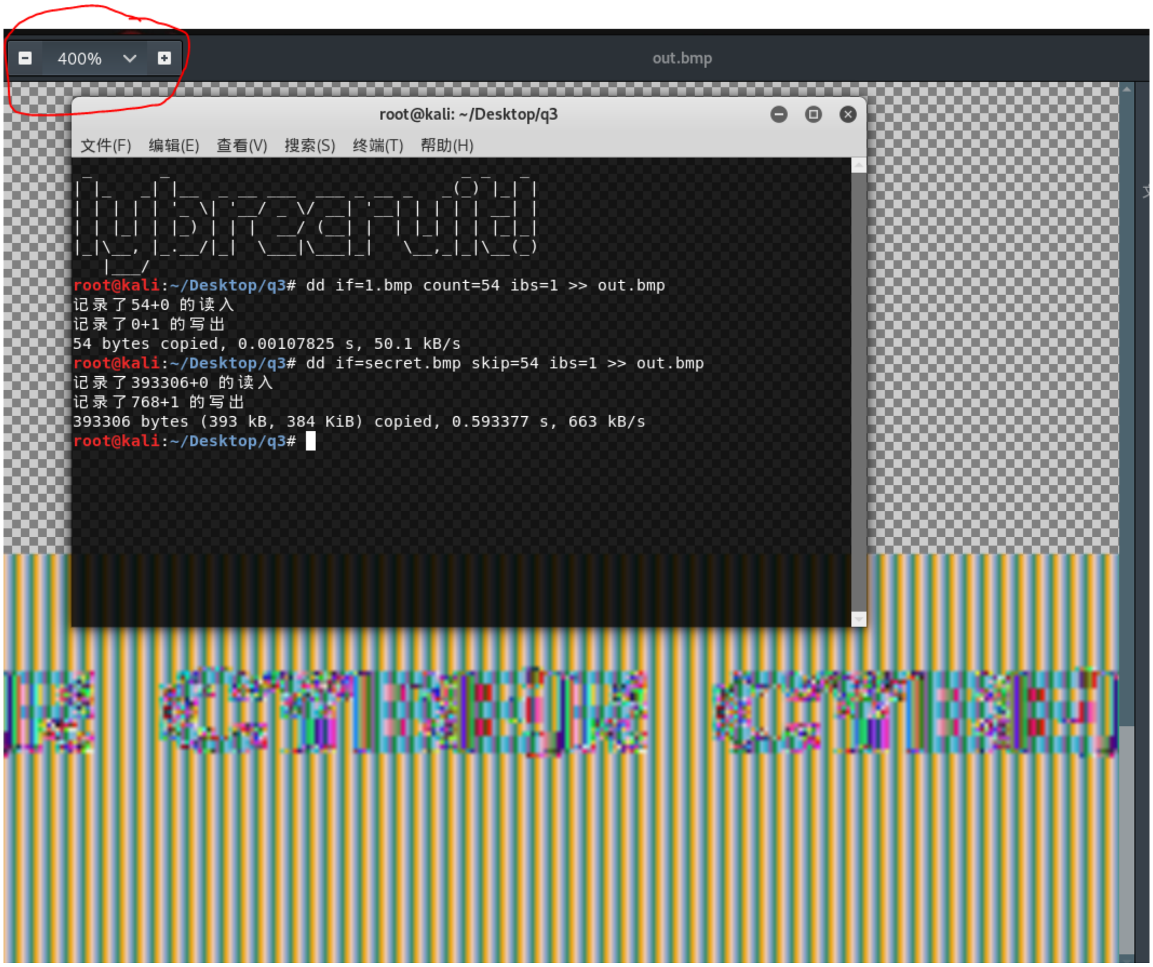
Unless the key is very large, it will cost more time (larger dictionary).

Q2.



```
>>> import binascii
>>> a = pow(int("4A070566DB88A19A8C1212E41
DCE3AE42112A8388DA3872EE44AF4C8E654A198",1
6),int("0D067636BAC6088AD2281E4BFFCACFEFEF
9BC1A69FB9E701063DFBAAB436E4C1",16),int("9
B51C20306EDE535C8FCAADBC3F3515E52A0D005703
DD449BEC66B23E2932313",16))
>>> binascii.unhexlify("%x" % a)
'Cowwanbanga!'
```

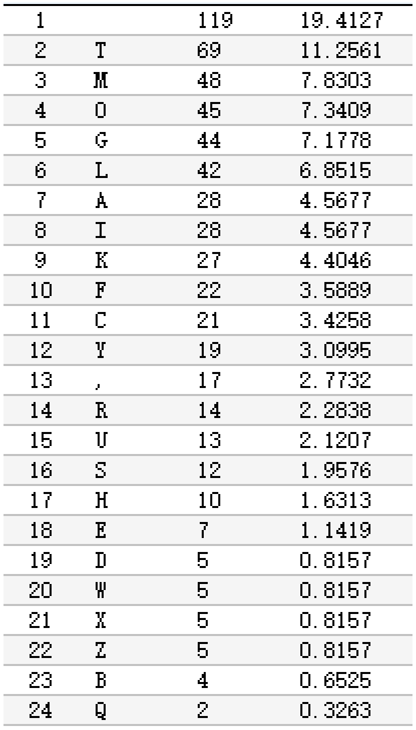
Q3.



```
 _        _                             _ _   _
| |_   _ | |__  _ __ ___  ___ _ __ _   _(_) |_| |
| | | | || '_ \| '__/ _ \/ __| '__| | | | | __| |
| | |_| || |_) | | |  __/ (__| |  | |_| | | |_|_|
|_|\__, ||_.__/|_|  \___|\___|_|   \__,_|_|\__(_)
   |___/
root@kali:~/Desktop/q3# dd if=1.bmp count=54 ibs=1 >> out.bmp
记录了54+0 的读入
记录了0+1 的写出
54 bytes copied, 0.00107825 s, 50.1 kB/s
root@kali:~/Desktop/q3# dd if=secret.bmp skip=54 ibs=1 >> out.bmp
记录了393306+0 的读入
记录了768+1 的写出
393306 bytes (393 kB, 384 KiB) copied, 0.593377 s, 663 kB/s
root@kali:~/Desktop/q3# 
```

Q4.

We calculate the frequency of any single letter: using website: (http://www.aihanyu.org/cncorpus/CpsTongji.aspx)



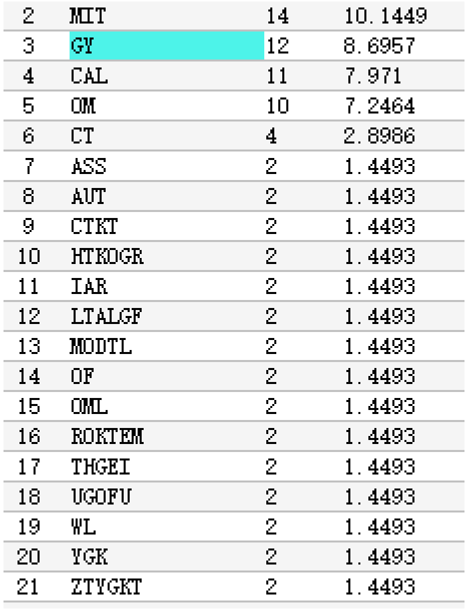
1		119	19.4127
2	T	69	11.2561
3	M	48	7.8303
4	O	45	7.3409
5	G	44	7.1778
6	L	42	6.8515
7	A	28	4.5677
8	I	28	4.5677
9	K	27	4.4046
10	F	22	3.5889
11	C	21	3.4258
12	Y	19	3.0995
13	,	17	2.7732
14	R	14	2.2838
15	U	13	2.1207
16	S	12	1.9576
17	H	10	1.6313
18	E	7	1.1419
19	D	5	0.8157
20	W	5	0.8157
21	X	5	0.8157
22	Z	5	0.8157
23	B	4	0.6525
24	Q	2	0.3263

.

The lecture shows that the most common letter in english is E.

In this scenes: E->T

And the most common letter: the



2	MIT	14	10.1449
3	GY	12	8.6957
4	CAL	11	7.971
5	OM	10	7.2464
6	CT	4	2.8986
7	ASS	2	1.4493
8	AUT	2	1.4493
9	CTKT	2	1.4493
10	HTKOGR	2	1.4493
11	IAR	2	1.4493
12	LTALGF	2	1.4493
13	MODTL	2	1.4493
14	OF	2	1.4493
15	OML	2	1.4493
16	ROKTEM	2	1.4493
17	THGEI	2	1.4493
18	UGOFU	2	1.4493
19	WL	2	1.4493
20	YGK	2	1.4493
21	ZTYGKT	2	1.4493

THE->MIT. This also confirmed e->t

t->m +19/-7

h->I +1/-27

e->t +15/-41

thus, there’s not Caesar cipher or vigenere cipher.

Maybe substitution cipher

Get through www.wordfrequency.info

Looking at the word: AW**th**GKO**t**O**e**L
Use online tools: http://www.hanginghyena.com/hangmansolver
try to search as “??th???t? e?”
get the answer: authorities
now we get

a->A
u->W
o->G
r->K
i->O
s->L

ZeYore

b->Z
f->Y

ResHair

d->R
p->H

epoEh

c->E

beSief

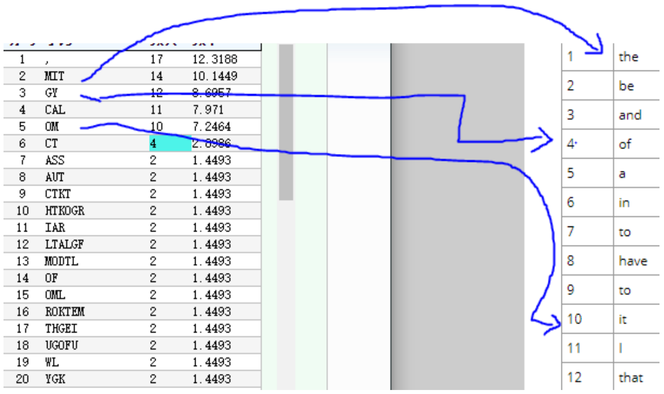
l->S

CisdoD

w->C
m->D

iFcredulitB

n->F
y->B
g->U
k->Q
v->X



1	,	17	12.3188
2	MIT	14	10.1449
3	GY	12	8.6957
4	CAL	11	7.971
5	OM	10	7.2464
6	CT	4	2.8986
7	ASS	2	1.4493
8	AUT	2	1.4493
9	CTKT	2	1.4493
10	HTKOGR	2	1.4493
11	IAR	2	1.4493
12	LTALGF	2	1.4493
13	MODTL	2	1.4493
14	OF	2	1.4493
15	OML	2	1.4493
16	ROKTEM	2	1.4493
17	THGEI	2	1.4493
18	UGOFU	2	1.4493
19	WL	2	1.4493
20	YGK	2	1.4493

1	the
2	be
3	and
4	of
5	a
6	in
7	to
8	have
9	to
10	it
11	I
12	that

according to existing keys, keep going, finally we can get:

it was the best of times, it was the worst of times, it was the age of wisdom, it was the age

of foolishness, it was the epoch of belief, it was the epoch of incredulity, it was the season of light, it was the season of darkness, it was the spring of hope, it was the winter of despair, we had everything before us, we had nothing before us, we were all going direct to heaven, we were all going direct the other way - in short, the period was so far like the present period, that some of its noisiest authorities insisted on its being received, for good or for evil, in the superlative degree of comparison only.

0	A	0	A	
1	B	25	Z	
2	C	4	E	
3	D	17	R	
4	E	19	T	
5	F	24	Y	
6	G	20	U	
7	H	8	I	
8	I	14	O	
9	J			
10	K	16	Q	
11	L	18	S	
12	M	3	D	
13	N	5	F	
14	O	6	G	
15	P	7	H	
16	Q			
17	R			
18	S	11	L	
19	T			
20	U	22	W	
21	V	23	X	
22	W	2	C	
23	X			
24	Y	1	B	
25	Z			

For encrypt: Y=(X+K) mod 26

0= 0+k mod 26 k= +-26

25 = 1+k mod 26 = 25 mod 26| -1 mod 26 => k = 24 | -2

4 = 2 + k mod 26 = 30mod26 | -22 mod 26 => k = 28 | -24

For decrypt: X= (Y - K) mod 26

0= 0-k mod 26 k= +-26

24=1-k mod 26 = 24mod26 | -2mod26=> k = -23 | 3

22 = 2-k mod 26 = 22mod26 | -4mod26=> k = -20 | 6

12=3-k mod 26=> k = -9 | 17

Sorry I can’t find the key. Maybe the key is “the=>mit”?

Q5.

Q6.