111550130 李慕庭 – Quiz1

Problem 1

(a) Frequency Analysis & Fill the table

Table 1: Ciphertext-to-plaintext mapping (ASCII 32–126)

Ciphertext	(space)	!	"	#	$	%	&	'	(	)	*	+	,	-	.
ASCII	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46
Plaintext	F	b		;	W	s		L	h		A		y	6	R
Ciphertext	/	0	1	2	3	4	5	6	7	8	9	:	;	<	=
ASCII	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61
Plaintext	n		G	c	(space)			t	1	M	i	&	B		z
Ciphertext	>	?	@	A	B	C	D	E	F	G	H	I	J	K	L
ASCII	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76
Plaintext	Π	S	o	,	H	d				u		N	j	'	C
Ciphertext	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[
ASCII	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91
Plaintext				T	P	—	I	e				v		O	k
Ciphertext	\	]	^	_	`	a	b	c	d	e	f	g	h	i	j
ASCII	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106
Plaintext		D				U	g	.	J	f				w	4
Ciphertext	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y
ASCII	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121
Plaintext	P	l		E	a		:		r		K	g			
Ciphertext	z	{	|	}	~										
ASCII	122	123	124	125	126										
Plaintext	x		Q	m											

(b) a=17, b=45

(c) If the encryption is based on substitution alphabets (for example, as 1-(b)), then the word “created” includes two “e”, so we can find all patterns in ciphertext form as ABCDECF (A, B, C, D, E, F are mutually different alphabets), assume it is a correct mapping and try to decryption. Besides, we know “e” is most frequent alphabets in usual (by Table 2 in 1-(a)), so we can reduce to a smaller possible mapping space. This method is frequency analysis with pattern matching.

(d) *a* need to coprime with 95, therefore, number of possible *a*

$$\phi(95) = \phi(5) * \phi(19) = (5-1) * (19-1) = 72$$

b need to within range [0, 94], therefore, number of possible *b* = 95

Total size of key space = 72 * 95 = 6840

In terms of nowadays’ computational ability, 6840 is a very small size. Moreover, by frequency analysis as we did in problem 1-(a), we can fix some mapping and therefore reduce the size of key space significantly. By

enumerating all possible keys and observing decryption result, we can find correct key easily.

(e) The size of key space of Monoalphabetic Substitution Cipher is the number of possible permutations over [32, 126] alphabets = 95!
95! , approximately equal to 10^{149} . Even, we can compute 10^{50} possibles in a second, we still need around 10^{92} years to enumerate them all !

(f) Two steps encryption

Step 1. Choose integers a, b, n such that $1 \leq a < n$, $\gcd(a, n) = 1$ and $0 \leq b < n$. Then, given plaintext x, we encrypt it in first step by

$$x' = (ax + b) mod\ n$$

Step 2. Use pseudo random generator to generate a key stream s, we encrypt output of first step by $y_i = x_i' \oplus s_i$ where y_i, x_i', s_i are i_{th}-bit of y, x', s respectively and the symbol $\oplus$ is bitwise XOR. y is ciphertext of x.

Criteria 1: This encryption consists of two steps encryption with different method.

Criteria 2: In the first step of encryption, the frequency doesn't change. However, in the second step, the same alphabet might be XOR by different bits and thus change the frequency of each alphabet.
In this way, make frequency analysis attacks more difficult.

Criteria 3: Decryption

Step 1: $x_i' = y_i' \oplus s_i$

Step 2: $x = a^{-1} * (x - b) mod\ n$

We can fully reverse ciphertext if we know a, b, n and s

Problem 2

(a) *a* is valid if a is coprime with *n* . →Number of valid *a* = $\phi(n)$

b has to within range [0, n-1] → Number of possible *b* = *n*

The total number of possible keys = $\phi(n) * n$

(b) Use the following table to list all elements

Elements have multiplicative inverse	Multiplicative inverse
1	1
7	13
11	11
13	7
17	23
19	19
23	17
29	29

```
def gcd(a, b):
    if a < b:
        a, b = b, a
    if b == 0:
        return a
    return gcd(b, a % b)
# (b)
for i in range(1, 30):
    if gcd(i, 30) == 1:
        for j in range(1, 30):
            if (i * j) % 30 == 1:
                print(i, j)
```

(Calculating Process)1

(c) a=37, b=58, n=97

```
for n in range(31, 100):
    for a in range(1, n):
        if gcd(a, n) != 1:
            continue
        for b in range(0, n):
            if(a * 81 + b) % n != 48:
                continue
            if(a * 14 + b) % n != 91:
                continue
            if(a * 3 + b) % n != 72:
                continue
            print(a, b, n)
```

(Searching Process)

(d) The way to decrypt affine encryption

$$y = ax + b \ (mod\ n)$$

$$x = a^{-1}(y - b) \ (mod\ n)$$

Therefore, $c = a^{-1} = 21$, $d = -a^{-1} * b = 43$

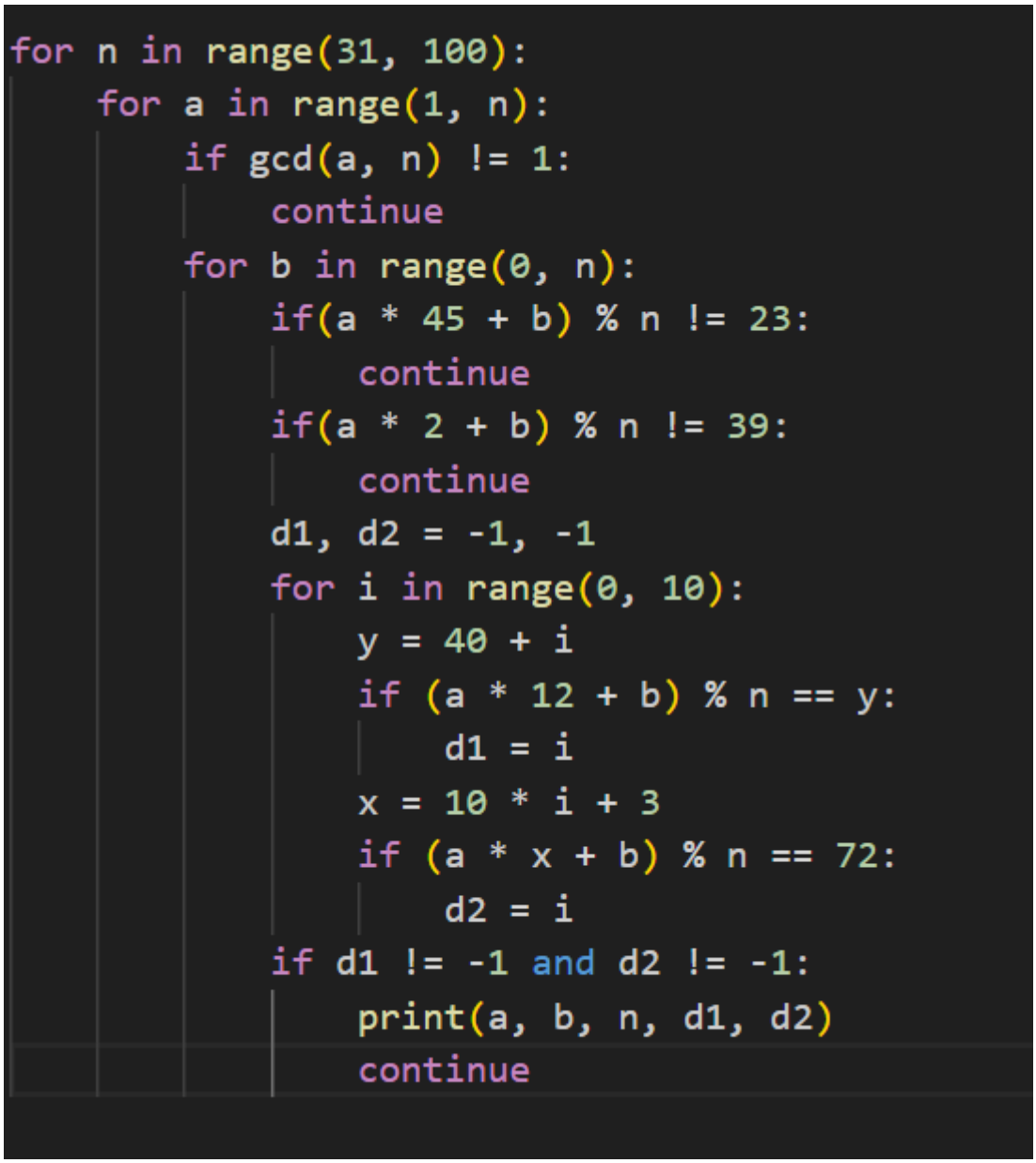
```
a, b, n = 37, 58, 97
inv_a = 0
for c in range(1, n):
    if(a * c) % n == 1:
        print(c)
        inv_a = c

d = inv_a * b % n
d = (n - d) % n
print(d)
```

(Calculating Process)

(e) a = 17, b = 5, n = 83

Missing digits are 3 and 4 in first and second row respectively.



```
for n in range(31, 100):
    for a in range(1, n):
        if gcd(a, n) != 1:
            continue
        for b in range(0, n):
            if(a * 45 + b) % n != 23:
                continue
            if(a * 2 + b) % n != 39:
                continue
            d1, d2 = -1, -1
            for i in range(0, 10):
                y = 40 + i
                if (a * 12 + b) % n == y:
                    d1 = i
                x = 10 * i + 3
                if (a * x + b) % n == 72:
                    d2 = i
            if d1 != -1 and d2 != -1:
                print(a, b, n, d1, d2)
                continue
```

(searching process)