COMP2700 Lab 7 Solutions

Exercise 1. The ciphertext below was encrypted using a substitution cipher. Decrypt the ciphertext without knowledge of the key.

```
lrvmnir bpr sumvbwvr jx bpr lmiwv yjeryrkbi jx qmbm wi
bpr xjvni mkd ymibrut jx irhx wi bpr riirkvr jx
ymbinlmtmipw utn qmumbr dj w ipmhh but bj rhnvwdmbr bpr
yjeryrkbi jx bpr qmbm mvvjudwko bj yt wkbrusurbmbwjk
lmird jk xjubt trmui jx ibndt
  wb wi kjb mk rmit bmiq bj rashmwk rmvp yjeryrkb mkd wbi
iwokwxwvmkvr mkd ijyr ynib urymwk nkrashmwkrd bj ower m
vjyshrbr rashmkmbwjk jkr cjnhd pmer bj lr fnmhwxwrd mkd
wkiswurd bj invp mk rabrkb bpmb pr vjnhd urmvp bpr ibmbr
jx rkhwopbrkrd ywkd vmsmlhr jx urvjokwgwko ijnkdhrii
ijnkd mkd ipmsrhrii ipmsr w dj kjb drry ytirhx bpr xwkmh
mnbpjuwbt lnb yt rasruwrkvr cwbp qmbm pmi hrxb kj djnlb
bpmb bpr xjhhjcwko wi bpr sujsru msshwvmbwjk mkd
wkbrusurbmbwjk w jxxru yt bprjuwri wk bpr pjsr bpmb bpr
riirkvr jx jqwkmcmk qmumbr cwhh urymwk wkbmvb
```

a) Compute the relative frequency of all letters a...z in the ciphertext. You may want to use a tool such as the open-source program JCrypTool for this task. However, a paper and pencil approach is also still doable.

b) Decrypt the ciphertext with the help of the relative letter frequency of the English language (see Table 1.1 in Sect. 1.2.2 in [Paar & Pelzl]). Note that the text is relatively short and that the letter frequencies in it might not perfectly align with that of general English language from the table.

Remark:

- *For your convenience, the cipher text in this question is provided as a separate file (cipher.txt), which you can download from Wattle.*
- *Deciphering the text can take some time; you are not expected to completely decipher the text during the lab session; but you are expected to be at least familiar with the frequency analysis technique used to solve this exercise.*

Answer:

a) Letter frequency analysis of the ciphertext:

letter	count	freq [%]	letter	count	freq [%]
A	5	0.77	N	17	2.63
B	68	10.53	O	7	1.08
C	5	0.77	P	30	4.64
D	23	3.56	Q	7	1.08
E	5	0.77	R	84	13.00
F	1	0.15	S	17	2.63
G	1	0.15	T	13	2.01
H	23	3.56	U	24	3.72
I	41	6.35	V	22	3.41
J	48	7.43	W	47	7.28
K	49	7.59	X	20	3.10
L	8	1.24	Y	19	2.94
M	62	9.60	Z	0	0.00

b) For this exercise, we use the JCryptool substitution analysis to decipher the ciphertext. See the provided tutorial video for JCryptool for details of how to perform substitution analysis. The plaintext for this exercise is the following (here we user the upper case letters to denote the plaintext to distinguish them from ciphertexts).

```
BECAUSE THE PRACTICE OF THE BASIC MOVEMENTS OF KATA IS THE FOCUS
AND MASTERY OF SELF IS THE ESSENCE OF MATSUBAYASHI RYU KARATE DO
I SHALL TRY TO ELUCIDATE THE MOVEMENTS OF THE KATA ACCORDING TO
MY INTERPRETATION BASED ON FORTY YEARS OF STUDY IT IS NOT AN EASY
TASK TO EXPLAIN EACH MOVEMENT AND ITS SIGNIFICANCE AND SOME MUST
REMAIN UNEXPLAINED TO GIVE A COMPLETE EXPLANATION, ONE WOULD HAVE
TO BE QUALIFIED AND INSPIRED TO SUCH AN EXTENT THAT HE COULD
REACH THE STATE OF ENLIGHTENED MIND CAPABLE OF RECOGNIZING
SOUNDLESS SOUND AND SHAPELESS SHAPE I DO NOT DEEM MYSELF THE
FINAL AUTHORITY BUT MY EXPERIENCE WITH KATA HAS LEFT NO DOUBT
THAT THE FOLLOWING IS THE PROPER APPLICATION AND INTERPRETATION I
OFFER MY THEORIES IN THE HOPE THAT THE ESSENCE OF OKINAWAN KARATE
WILL REMAIN INTACT
```

This is an excerpt from The Essence of Okinawan Karate-Do by Shoshin Nagamine, Tuttle Publishing, 1998.

Exercise 2. We received the following ciphertext which was encoded with a shift cipher:

```
xultpaajcxitltlxaarpjhtiwtgxktghidhipxciwtvgtpilpitghlxiwiwtxgqadds
```

Perform an attack against the cipher based on a letter frequency count: How many letters do you have to identify through a frequency count to recover the key? What is the plaintext?

Answer:

Recall that in a shift cipher, we first encode the alphabet into numbers, mapping the letter ‘a’ to 0, ‘b’ to 1, and so on. The encryption and the decryption functions for the shift cipher for each character are defined as, respectively:

$$y = e_k(x) \equiv x + k \, mod \, 26$$

$$x = d_k(y) \equiv y - k \, mod \, 26$$

where x is the plaintext, y is the the ciphertext, and k is the key, i.e., the amount of shift that was applied to the plaintext. To decipher the given ciphertext we need to discover the key k.

The key is uniquely determined by one letter pair of plaintext-ciphertext. The frequency count of the most frequent ciphertext letters yields:

- # t = 10
- # x = 7
- # a = 5
- # p = 5

So it is likely that the plaintext letter ‘e’ is mapped to ciphertext letter ‘t’. The corresponding encodings of the letters ‘e’ and ‘t’ are 4 and 19, respectively. By the definition of the encryption function above, we have: x=4, y=19, and

$y = 19 = e_k(x) \equiv e_k(4) \equiv 4 + k \, mod \, 26$
$\Rightarrow \; 19 - 4 \equiv k \, mod \, 26$
$\Rightarrow \; k \equiv 15 \, mod \, 26.$

Knowing the key k, we can now decrypt the entire text. For example, the letter ‘x’, which is encoded as the number 23, can now be decrypted as follows:

$$d_k(23) = 23 - 15 \equiv 8 \, mod \, 26$$

Since 8 is the encoding of ‘i’, we know that ‘x’ decrypts to ‘i’. Repeating this process to the remaining letters, we obtain the following plaintext:

```
ifweallunitewewillcausetheriverstostatinthegreatwaterswiththeirblood
```

The original text, with punctuations reads:

If we all unite, we will cause the rivers to stain the great waters with their blood.

This is an excerpt from: Native American Testimony by Peter Nabokov, Penguin, 1991.

You can also automate the decryption using JCryptool, under the menu Algorithms -> Classic -> Caesar.

Exercise 3. As we learned in this chapter, modular arithmetic is the basis of many crypto systems. As a consequence, we will address this topic with several problems in this and upcoming chapters. Let's start with an easy one: Compute the result without a calculator.

a) 15·29 mod 13
b) 2·29 mod 13
c) 2·3 mod 13
d) −11·3 mod 13

The results should be given in the range from 0,1,..., modulus-1. Briefly describe the relation between the different parts of the problem.

Answer:

We use the property that $a \times b \; mod \; c = (a \; mod \; c) \times \; (b \; mod \; c) \; mod \; c$. Note that in computing $a \; mod \; b$, when a is negative, you can work with its positive equivalence, e.g., by adding multiple of b's to a until it becomes positive. For example, $-11 \; mod \; 13 = (-11 + 13 \cdot 1) \; mod \; 13 = 2 \; mod \; 13$.

a) $15 \cdot 29 \; mod \; 13 \equiv 2 \cdot 3 \; mod \; 13 \equiv 6 \; mod \; 13$
b) $2 \cdot 29 \; mod \; 13 \equiv 2 \cdot 3 \; mod \; 13 \equiv 6 \; mod \; 13$
c) $2 \cdot 3 \; mod \; 13 \equiv 6 \; mod \; 13$
d) $-11 \cdot 3 \; mod \; 13 \equiv \; 2 \cdot 3 \; mod \; 13 \equiv 6 \; mod \; 13$.

Exercise 4. Construct the addition and the multiplication table for the rings Z_5 and Z_6. There are elements in Z_6 without a multiplicative inverse. Which elements are these. Why does a multiplicative inverse exist for all nonzero elements of Z_5.

Answer:

We will explain how to create the addition and multiplication tables for ring Z_5. You can apply this technique to do the same for Z_6.

We can recall that the ring Z_5 contains 5 elements: {0,1,2,3,4}. There are two operations on this ring:

- addition
- multiplication

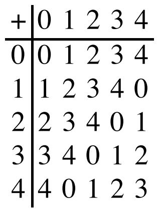
Both operations work on two elements from the ring Z_5 and return an element within the ring Z_5. In this exercise, we will create the addition and multiplication tables for the ring Z_5. That is, we will add/times all possible pairs of elements from Z_5 together and then take the result modulus 5 (so that the resulting value is an element in the ring Z_5). We can visualise how this works below:

+	**0**	**1**	**2**	**3**	**4**
0	$0 + 0 \; mod \; 5$	$0 + 1 \; mod \; 5$	$0 + 2 \; mod \; 5$	$0 + 3 \; mod \; 5$	$0 + 4 \; mod \; 5$
1	$1 + 0 \; mod \; 5$	$1 + 1 \; mod \; 5$	$1 + 2 \; mod \; 5$	$1 + 3 \; mod \; 5$	$1 + 4 \; mod \; 5$
2	$2 + 0 \; mod \; 5$	$2 + 1 \; mod \; 5$	$2 + 2 \; mod \; 5$	$2 + 3 \; mod \; 5$	$2 + 4 \; mod \; 5$
3	$3 + 0 \; mod \; 5$	$3 + 1 \; mod \; 5$	$3 + 2 \; mod \; 5$	$3 + 3 \; mod \; 5$	$3 + 4 \; mod \; 5$
4	$4 + 0 \; mod \; 5$	$4 + 1 \; mod \; 5$	$4 + 2 \; mod \; 5$	$4 + 3 \; mod \; 5$	$4 + 4 \; mod \; 5$

x	**0**	**1**	**2**	**3**	**4**
0	$0 * 0\ mod\ 5$	$0 * 1\ mod\ 5$	$0 * 2\ mod\ 5$	$0 * 3\ mod\ 5$	$0 * 4\ mod\ 5$
1	$1 * 0\ mod\ 5$	$1 * 1\ mod\ 5$	$1 * 2\ mod\ 5$	$1 * 3\ mod\ 5$	$1 * 4\ mod\ 5$
2	$2 * 0\ mod\ 5$	$2 * 1\ mod\ 5$	$2 * 2\ mod\ 5$	$2 * 3\ mod\ 5$	$2 * 4\ mod\ 5$
3	$3 * 0\ mod\ 5$	$3 * 1\ mod\ 5$	$3 * 2\ mod\ 5$	$3 * 3\ mod\ 5$	$3 * 4\ mod\ 5$
4	$4 * 0\ mod\ 5$	$4 * 1\ mod\ 5$	$4 * 2\ mod\ 5$	$4 * 3\ mod\ 5$	$4 * 4\ mod\ 5$

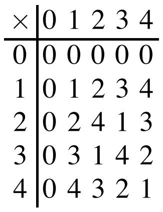
Below are the resulting addition and multiplication tables.

Addition table for Z_5



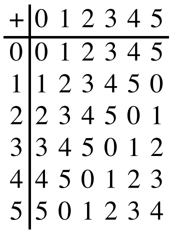
+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

Multiplication table for Z_5



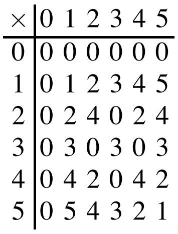
×	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

Addition table for Z_6



+	0	1	2	3	4	5
0	0	1	2	3	4	5
1	1	2	3	4	5	0
2	2	3	4	5	0	1
3	3	4	5	0	1	2
4	4	5	0	1	2	3
5	5	0	1	2	3	4

Multiplication table for Z_6



×	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	1	2	3	4	5
2	0	2	4	0	2	4
3	0	3	0	3	0	3
4	0	4	2	0	4	2
5	0	5	4	3	2	1

For the multiplicative inverse of an element a in Z_5 to exist, we must have that $\gcd(a, 5) = 1$. Since 5 is prime, we have that a multiplicative inverse exists for all nonzero elements. This is because all nonzero elements smaller than 5 are relatively prime to 5. For example, $\gcd(4,5) = 1$. However, $\gcd(0,5) = 5$ and so 0 doesn't have a multiplicative inverse in Z_5.

For the multiplicative inverse of an element a in Z_6 to exist, we must have that $\gcd(a, 6) = 1$. Elements without a multiplicative inverse in Z_6 are 2, 3, 4 and 0. For example, $\gcd(4,6) = 2$.

Exercise 5. Compute without a calculator:

a) 1/5 mod 13
b) 1/5 mod 7
c) 3·2/5 mod 7

Answer:

Let us do a working example for a) to show how we can compute the multiplicative inverse of $5\ mod\ 13$. Here we can see we are dealing with the ring Z_{13}.

We want to compute $1/5\ mod\ 13$ = $1 * 5^{-1}\ mod\ 13$

Thus, we need to calculate $5^{-1}\ mod\ 13$ (the multiplicative inverse of $5\ mod\ 13$).

$\gcd(5,13) = 1$ so we know that this multiplicative inverse exists.

We also know that in general for an element a in a ring Z_m, where its multiplicative inverse exists,

$$a * a^{-1} = 1\ mod\ m$$

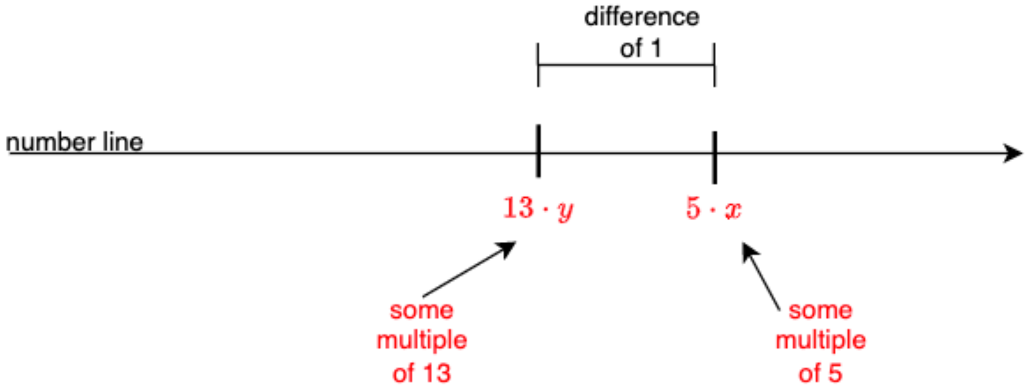
and so we know that

$$5 * 5^{-1} = 1\ mod\ 13$$

Let $x = 5^{-1}$

$$5 * x = 1\ mod\ 13$$

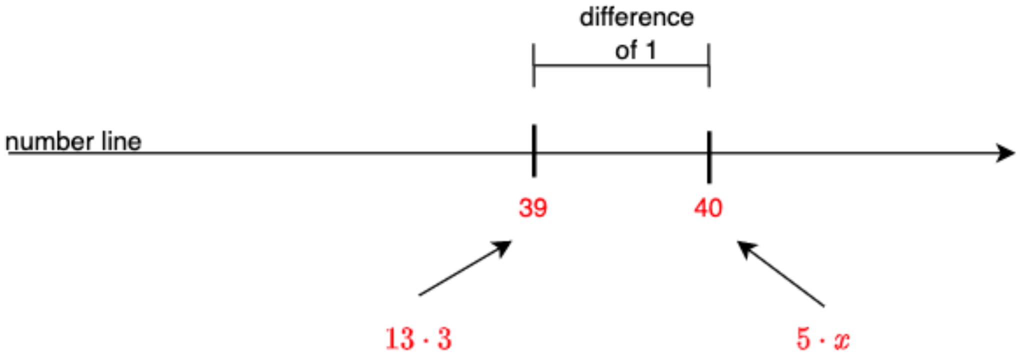
Now if we draw out a number line, we should see that some multiple of 13 will be 1 smaller than some multiple of 5



Let us look through multiples of 13 until we find one that is 1 less than a multiple of 5

13, 26, **39**

Great! We have found $39 = 13 * 3$ which is 1 less than 40 (a multiple of 5)!



This implies that $40 = 5 * x \quad \Rightarrow \quad x = 8 \quad \Rightarrow \quad 5^{-1} = 8$

Now we can substitute 5^{-1} back into our original problem

$1 * 5^{-1} \, mod \, 13 = 1 * 8 \, mod \, 13 = 8$

The other answers are below:

a) 8
b) 3, since $3 \cdot 5 \, mod \, 7 \equiv 1 \, mod \, 7$
c) 4, since $3 \cdot \frac{2}{5} \, mod \, 7 \equiv 3 \cdot 2 \cdot 5^{-1} \, mod \, 7 \equiv 3 \cdot 2 \cdot 3 \, mod \, 7 \equiv 4 \, mod \, 7$.

Exercise 6.

This problem deals with the affine cipher with the key parameters *a* = 7, *b* = 22. Decrypt the text below:

```
falszztysyjzyjkywjrztyjztyynaryjkyswarztyegyyj
```

Answer:

We first map letters to numbers, for example, ‘a’ maps to 0, ‘b’ maps to 1, ‘c’ maps to 2, and so on. Recall that the decryption function for the affine cipher with key (a,b) and modulus m is

$$d_k(y) = (y - b)a^{-1} \, mod \, m$$

The inverse of a is $a^{-1} \equiv 7^{-1} \equiv 15 \, mod \, 26$. (*Note: Since the modulus is quite small in this case, we can simply iterate through all possible values between 2 to 25 to find the inverse of a. For large modulus, we will need a more efficient algorithm such as the Euclidean algorithm*).

So for example, the letter ‘f’ (which maps to 5) should be decrypted as:

$$d_k(5) = (5 - 22) \times 7^{-1} \, mod \, 26 = 5$$

That is, ‘f’ actually decrypts to ‘f’. Repeating this for other letters, we obtain the plaintext:

```
FIRST THE SENTENCE AND THEN THE EVIDENCE SAID THE QUEEN
```

(capitalised the letters and added spaces for clarity)

Exercise 7.

In an attack scenario, we assume that the attacker Oscar manages somehow to provide Alice with a few pieces of plaintext that she encrypts. Show how Oscar can break the affine cipher by using two pairs of plaintext–ciphertext, (x_1, y_1) and (x_2, y_2). What is the condition for choosing x_1 and x_2?

Remark: *In practice, such an assumption turns out to be valid for certain settings, e.g., encryption by Web servers, etc. This attack scenario is, thus, very important and is denoted as a chosen plaintext attack.*

Answer:

Suppose the key is (a,b) and the modulus is m. Then we have

$$y_1 \equiv a \cdot x_1 + b \, mod \, m$$

$$y_2 \equiv a \cdot x_2 + b \, mod \, m$$

Substitution the lower equation from the upper, we get

$(y_1 - y_2) \equiv a \cdot (x_1 - x_2) \, mod \, m$

from which we can derive a:

$$a \equiv (x_1 - x_2)^{-1} \, (y_1 - y_2) \, mod \, m.$$

The inverse of $(x_1 - x_2)$ must exist modulo m, i.e., $\gcd\big((x_1 - x_2), m\big) = 1$.

From this and the first equation we derive b:

$b \equiv y_1 - ax_1 \text{ mod } m$

Extension Exercises -- Optional

Exercise 8 (*).

We consider the long-term security of the Advanced Encryption Standard (AES) with a key length of 128-bit with respect to exhaustive key-search attacks. AES is perhaps the most widely used symmetric cipher at this time.

a) Assume that an attacker has a special purpose application specific integrated circuit (ASIC) which checks $5 \cdot 10^8$ keys per second, and she has a budget of \$1 million. One ASIC costs \$50, and we assume 100% overhead for integrating the ASIC (manufacturing the printed circuit boards, power supply, cooling, etc.). How many ASICs can we run in parallel with the given budget? How long does an average key search take? Relate this time to the age of the Universe, which is about 10^{10} years.

b) We try now to take advances in computer technology into account. Predicting the future tends to be tricky but the estimate usually applied is Moore's Law, which states that the computer power doubles every 18 months while the costs of integrated circuits stay constant. How many years do we have to wait until a key-search machine can be built for breaking AES with 128 bit with an average search time of 24 hours? Again, assume a budget of \$1 million (do not take inflation into account).

Hint: This is an instance of linear search. In this case, the underlying assumption is that keys are uniformly distributed so on average, given a key space of n, is (n+1)/2. Asymptotically this approaches n/2. See, e.g.,

https://en.wikipedia.org/wiki/Linear_search#Analysis

Answer:

One search engine costs \$ 100 including overhead. Thus, 1 million dollars buy us 10,000 engines.

a) With 10,000 engines, each engine is capable of checking $5 \cdot 10^8$ keys per second, we can test $5 \cdot 10^8 \cdot 10^4 = 5 \cdot 10^{12}$ keys per second. We have 2^{128} possible keys to check. Since this is an instance of linear search, on average, we may have to check $\sqrt{2^{128}} = 2^{127}$ keys. This will take:

$(2^{127}$keys$)/(5 \cdot 10^{12}$keys/sec$) = 3.40 \cdot 10^{25}$sec $= 1.08 \cdot 10^{18}$years

That is about 10^8 = 100, 000, 000 times longer than the age of the universe. Good luck.

b) Assuming each year has 365 days.

- Without any improvement in computer power, the key search will take $1.08 \cdot 10^{18} \cdot 365$ days.
- Let i denotes the number of iterations of the Moore's Law. For example, if $i = 1$ then the computer power increases by $2^i = 2^1 = 2$ (i.e., it's twice as fast). If $i = 2$ then the computer power increases by $2^2 = 4$ (i.e., it's four times as fast), and so on. Let us first determine how many iterations of Moore's Law are required to bring key search time to 1 day. This is given by the time required to search the key, divided by 2^i , giving us the equation:

$$\frac{1.08 \cdot 10^{18} \cdot 365}{2^i} = 1\ day$$
$$\Rightarrow\ 2^i = 1.08 \cdot 10^{18} \cdot 365$$
$$\Rightarrow\ \ i = \log_2(1.08 \cdot 10^{18} \cdot 365) = 68.42$$

We round this number up to 69 assuming the number of Moore iterations is discrete. So we need 69 iterations of Moore's Law. From the assumption of the exercise, we have that each iteration takes 18 months, or equivalently, 1.5 years. Thus, we have to wait for:
$1.5 \cdot 69 = 103.5$ years
Note that it is extremely unlikely that Moore's Law will be valid for such a time period! Thus, a 128 bit key seems impossible to brute-force, even in the foreseeable future.

Exercise 9 (*).

An obvious approach to increase the security of a symmetric algorithm is to apply the same cipher twice, i.e.:

$$y = e_{k2}(e_{k1}(x))$$

As is often the case in cryptography, things are very tricky, and results are often different from the expected and/or desired ones. In this problem we show that a double encryption with the affine cipher is only as secure as single encryption! Assume two affine ciphers

$e_{k1} = a_1x + b_1\ mod\ 26$ and $e_{k2} = a_2x + b_2\ mod\ 26$.

a) Show that there is a single affine cipher $e_{k3} = a_3x + b_3\ mod\ 26$ which performs exactly the same encryption (and decryption) as the combination $e_{k2}(e_{k1}(x))$.

b) Find the values for a_3, b_3 when $a_1 = 3, b_1 = 5$ and $a_2 = 11, b_2 = 7$.

c) For verification: (1) encrypt the letter K first with e_{k1} and the result with e_{k2}, and (2) encrypt the letter K with e_{k3}.

d) Briefly describe what happens if an exhaustive key-search attack is applied to a double-encrypted affine ciphertext. Is the effective key space increased?

Remark: *The issue of multiple encryption is of great practical importance in the case of the Data Encryption Standard (DES), for which multiple encryption (in particular, triple encryption) does increase security considerably.*

Answer:

a) Expanding the equation for the encryption $e_{k2}(e_{k1}(x))$ we have:

$$e_{k2}\big(e_{k1}(x)\big) = a_2(a_1x + b_1) + b_2 = (a_1a_2)x + (b_1a_2 + b_2) \equiv a_3x + b_3 \; mod \; 26$$

The last term $a_3x + b_3$, where $a_3 \equiv a_1a_2 \; mod \; 26$ and $b_3 \equiv b_1a_2 + b_2 \; mod \; 26$, is the same as the encryption function with key (a_3, b_3). Composing two affine ciphers yields another affine cipher, so there is no improvement in terms of security.

b) $a_3 = a_1a_2 = 3 \cdot 11 = 33 \equiv 7 \; mod \; 26$.
$b_3 = b_1a_2 + b_2 = 5 \cdot 11 + 7 = 62 \equiv 10 \; mod \; 26$.

c) The letter ‘K’ is encoded as 10 (the 11th alphabet, counting from 0).
$y_1 = e_{k1}(10) = 3 \cdot 10 + 5 = 35 = 9 \; mod \; 26$ (the letter ‘J’)
$e_{k2}(y_1) = 11 \cdot 9 + 7 = 106 \equiv 2 \; mod \; 26$ (the letter ‘C’)

$e_{k3}(10) = 7 \cdot 10 + 10 = 80 \equiv 2 \; mod \; 26$. (the letter ‘C’)

d) No, the key space is not affected; there are still only 312 possible keys.