A Short Introduction to Cryptography

Brandon Bate

1 Very Basic Ciphers

Alice and Bob are friends. They like to send messages to each other. Unfortunately, Eve is always trying to intercept and read their messages (she eavesdrops). Alice and Bob do not appreciate this. One solution to this problem is for Alice to directly hand her messages to Bob. This will prevent Eve from ever seeing the message. However, this form of message transportation is often times impractical, especially if Alice and Bob are far away from each other. Another solution is for Alice to convert her message to a *ciphertext* by using a *cipher*. A cipher is simply a method of converting her message, which we will refer to as the *plaintext*, into a series of symbols in such a way that the original meaning of the message is obscured. This process of converting a plaintext to ciphertext is called *encryption*. When Bob receives the ciphertext, he can then *decrypt* the ciphertext to recover the original plaintext. Of course, Alice and Bob keep their cipher method a secret from Eve. Thus (theoretically) even if Eve intercepts the ciphertext message, she will be unable to ascertain the original plaintext.

Ciphers have been used throughout much of history, especially in military conflicts. Julius Caesar had a particularly simple cipher which he used to convey military secrets. This cipher is now commonly known as the *Caesar cipher*. Suppose Caesar wishes to encrypt the following message:

```
i came, i saw, i conquered.
```

We refer to this message as the plaintext. We shall always write the plaintext in lower-case letters. To encrypt a message, Caesar would shift each letter of the alphabet forward by 3 letters:

```
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
↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓
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
```

He would then us the above rule to convert his plaintext into the following ciphertext:

```
L FDPH, L VDZ, L FRQTXHUHG.
```

We shall always write our ciphertext in upper-case to distinguish it from the plaintext.

Often times, spaces and punctuation are not included in a ciphertext. This makes the message harder for Eve to decode yet still discernible to Bob. If we were to leave out punctuation in our Caesar example then we'd obtain the following ciphertext instead:

```
LFDPHLVDZLFRQTXHUHG.
```

To decrypt a message, the receiver simply shifts back by 3 letters:

```
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
↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓
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
```

Applying this rule to our ciphertext yields our original plaintext message:

```
i came, i saw, i conquered.
```

Exercise 1.1. Encrypt the following message using the Caesar cipher:

```
experience is the teacher of all things
```

Exercise 1.2. Decrypt the following message using the Caesar cipher:

```
LORYHWKHQDPHRIKRQRU
```

We can generalize the Caesar cipher by shifting forward by k letters where k is an integer between 1 and 25. Such ciphers are known as *shift ciphers*. The Caesar cipher is simply a shift cipher with $k = 3$. The integer k is referred to as the *key* for the shift cipher. When sending a message with the shift cipher, both Alice and Bob must agree upon a common key to use.

Suppose Alice wishes to send the following plaintext message to Bob:

```
hello world
```

She decides to use a shift cipher with $k = 12$ and makes sure that Bob knows which key she is using. When we shift forward by $k = 12$ letters, we obtain the following rule:

```
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
↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓
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
```

When we apply this rule we obtain the following ciphertext:

```
TQXXA IADXP
```

Since Bob knows that a shift of $k = 12$ was used, he can write down the same rule Alice did. To decrypt, he simply locates the appropriate ciphertext letter in the bottom row and follows the arrow backwards to the corresponding plaintext letter. Bob will recover the original plaintext when he does this.

Exercise 1.3. The following ciphertext was computed using the shift cipher with $k = 7$:

```
NV WHAYPVAZ
```

Decrypt this message.

The shift cipher is not considered a very strong cipher. Even if Eve does not know the key used, she can attempt to decrypt the ciphertext by "decrypting" for each $k = 1, \ldots, 25$. For most choices of k she will obtain gibberish as her "plaintext", but for at least one of these k, she will obtain an intelligible message. Obviously, this intelligible message was Alice's original plaintext.

Exercise 1.4. Eve has intercepted the following ciphertext which was created by using a shift cipher:

```
LQNNBN
```

Decrypt this message.

Exercise 1.5. Eve has intercepted the following ciphertext which was created by using a shift cipher:

```
CNMNBYQLIHANLIOMYLM
```

Decrypt this message.

The following problem was taken from *Introduction to Cryptography and Coding Theory* by Trappe and Washington.

Exercise 1.6. Caesar wants to arrange a secret meeting with Marc Antony, either at the Tiber (the river) or at the Coliseum (the arena). He sends the ciphertext:

```
EVIRE
```

The message was encrypted using a shift cipher; however, Antony does not know the key, so he tries all possibilities. Where will he meet Caesar?

We have seen that the shift cipher has very few keys, which makes the shift cipher a very weak encryption methods. For Eve to decrypt a message, all she had to do was attempt to decrypt the message with each possible key. Such an "attack" on a cipher is called a *brute force attack*. Next we discuss the *Substitution cipher*, which has the advantage of having many keys, making a brute force attack much more difficult.

Before encrypting a message using the substitution cipher, Alice and Bob must agree upon a common key. A key for the substitution cipher is simply a permutation of the alphabet such as the following:

```
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
↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓
D I S K E O W T V A H N Q X M C L R F P G Z U B Y J
```

Using this particular key, Alice would encrypt the message:

```
eve could never break this cipher
```

as

```
EZE SMGNK XEZER IREDH PTVF SVCTER
```

The substitution cipher has $26! = 4.03291461 \times 10^{26}$ possible keys. This makes a brute force attack on the substitution cipher impractical. Although this does make the substitution cipher harder to break than the shift cipher, the substitution cipher is still a relatively weak encryption method (as we shall soon see). Another drawback to the substitution cipher is that the keys for the algorithm can be long and complicated. It is in this regard that the shift cipher has an advantage over the substitution cipher.

In the English language, not all letters are used equally often. Below is a table giving the overall frequencies of each letter in the alphabet.

Letter	Frequency
a	8.167%
b	1.492%
c	2.782%
d	4.253%
e	12.702%
f	2.228%
g	2.015%
h	6.094%
i	6.966%
j	0.153%
k	0.772%
l	4.025%
m	2.406%

Letter	Frequency
n	6.749%
o	7.507%
p	1.929%
q	0.095%
r	5.987%
s	6.327%
t	9.056%
u	2.758%
v	0.978%
w	2.360%
x	0.150%
y	1.974%
z	0.074%

We shall use this table to break the substitution cipher. Such an attack is called *frequency analysis*.

Exercise 1.7. Eve has obtained the following ciphertext (we have included spacing and punctuation to make this problem simpler, but the same techniques we will use apply for ciphertexts without spacing or punctuation):

```
FO QEN SYPV REOY EIB YSYPVLIY MEB RYHO OMY AEHY YCAYXO EI LRB KEI QML NEO FI
OMY NMEBLQ OMY RYESYN LH OMY OPYY KEBY ETEFINO OMY YRYAOPFA RFTMO. FI OMY BEV
OFKY OMY NOPYYO QEN BJNOV, GJO EO IFTMO OMY BYQ NYOORYB OMY BJNO EIB OMY LRB
KEI RFDYB OL NFO REOY GYAEJNY MY QEN BYEH EIB ILQ EO IFTMO FO QEN ZJFYO EIB MY
HYRO OMY BFHHYPYIAY.
```

This message was encrypted using the substitution cipher. Decrypt this message.

We begin by counting the number of times each ciphertext character appears:

Letter	Count	Frequency
A	6	2.308%
B	17	6.538%
C	1	0.385%
D	1	0.385%
E	26	10.000%
F	14	5.385%
G	2	0.769%
H	7	2.692%
I	15	5.769%
J	5	1.923%
K	4	1.538%
L	8	3.077%
M	19	7.308%

Letter	Count	Frequency
N	14	5.384%
O	38	14.615%
P	6	2.308%
Q	8	3.077%
R	11	4.231%
S	3	1.154%
T	3	1.154%
U	0	0.000%
V	4	1.538%
W	0	0.000%
X	1	0.385%
Y	45	17.308%
Z	1	0.385%

Since e, t, and a are the most common letters in the English language, this table suggests that perhaps `e` $\to$ `Y`, `t` $\to$ `O`, and `a` $\to$ `E`. We substitute these plaintext letters in to our ciphertext:

```
Ft QaN SePV Rate aIB eSePVLIe MaB ReHt tMe AaHe eCAeXt aI LRB KaI QML Nat FI
tMe NMaBLQ tMe ReaSeN LH tMe tPee KaBe aTaFINt tMe eReAtPFA RFTMt.  FI tMe BaV
tFKe tMe NtPeet QaN BJNtV, GJt at IFTMt tMe BeQ NettReB tMe BJNt aIB tMe LRB
KaI RFDeB tL NFt Rate GeAaJNe Me QaN BeaH aIB ILQ at IFTMt Ft QaN ZJFet aIB Me
HeRt tMe BFHHePeIAe.
```

Notice how the word tMe occurs often. This suggests that h → M. Also, the first word is Ft. We have already guessed that a → Y and since *at* and *it* are the only two letter words ending in *t*, we suspect that i → F. When we substitute these plaintext letters in to our ciphertext we get:

```
it QaN SePV Rate aIB eSePVLIe haB ReHt the AaHe eCAeXt aI LRB KaI QhL Nat iI
the NhaBLQ the ReaSeN LH the tPee KaBe aTaiINt the eReAtPiA RiTht.  iI the BaV
tiKe the NtPeet QaN BJNtV, GJt at IiTht the BeQ NettReB the BJNt aIB the LRB
KaI RiDeB tL Nit Rate GeAaJNe he QaN BeaH aIB ILQ at IiTht it QaN ZJiet aIB he
HeRt the BiHHePeIAe.
```

The words RiTht and IiTht both appear. This suggests that g → T. Substituting in for the plaintext character yields:

```
it QaN SePV Rate aIB eSePVLIe haB ReHt the AaHe eCAeXt aI LRB KaI QhL Nat iI
the NhaBLQ the ReaSeN LH the tPee KaBe agaiINt the eReAtPiA Right.  iI the BaV
tiKe the NtPeet QaN BJNtV, GJt at Iight the BeQ NettReB the BJNt aIB the LRB
KaI RiDeB tL Nit Rate GeAaJNe he QaN BeaH aIB ILQ at Iight it QaN ZJiet aIB he
HeRt the BiHHePeIAe.
```

Notice the word agaiINt appears. This suggests that n → I and s → N. When we substitute these plaintext letters in to our ciphertext we get:

```
it Qas SePV Rate anB eSePVLne haB ReHt the AaHe eCAeXt an LRB Kan QhL sat in
the shaBLQ the ReaSes LH the tPee KaBe against the eReAtPiA Right.  in the BaV
tiKe the stPeet Qas BJstV, GJt at night the BeQ settReB the BJst anB the LRB
Kan RiDeB tL sit Rate GeAaJse he Qas BeaH anB nLQ at night it Qas ZJiet anB he
HeRt the BiHHePenAe.
```

Notice that the word anB appears. This suggests that d → B. Also notice that the word stPeet appears. This suggests that r → P. When we substitute these plaintext letters in to our ciphertext we get:

```
it Qas SerV Rate and eSerVLne had ReHt the AaHe eCAeXt an LRd Kan QhL sat in
the shadLQ the ReaSes LH the tree Kade against the eReAtriA Right.  in the daV
tiKe the street Qas dJstV, GJt at night the deQ settRed the dJst and the LRd
Kan RiDed tL sit Rate GeAaJse he Qas deaH and nLQ at night it Qas ZJiet and he
HeRt the diHHerenAe.
```

The words Qas and deQ suggest that w → Q. The word dJst suggests that u → J. The word tL suggests that o → L. When we substitute these plaintext letters in to our ciphertext we get:

it was SerV Rate and eSerVone had ReHt the AaHe eCAeXt an oRd Kan who sat in the shadow the ReaSes oH the tree Kade against the eReAtriA Right. in the daV tiKe the street was dustV, Gut at night the dew settRed the dust and the oRd Kan RiDed to sit Rate GeAause he was deaH and now at night it was Zuiet and he HeRt the diHHerenAe.

The word dustV suggests that y → V. The word Gut suggests that b → G. The word oRd suggests that l → R. When we substitute these plaintext letters in to our ciphertext we get:

it was Sery late and eSeryone had leHt the AaHe eCAeXt an old Kan who sat in the shadow the leaSes oH the tree Kade against the eleAtriA light. in the day tiKe the street was dusty, but at night the dew settled the dust and the old Kan liDed to sit late beAause he was deaH and now at night it was Zuiet and he Helt the diHHerenAe.

The word eSeryone suggests that v → S. The word leHt suggests that f → H. The word Zuiet suggests that q → Z. From the word Kan and the context of the sentence, we suspect that m → K. The word beAause suggests that c → A. When we substitute these plaintext letters in to our ciphertext we get:

it was very late and everyone had left the cafe eCceXt an old man who sat in the shadow the leaves of the tree made against the electric light. in the day time the street was dusty, but at night the dew settled the dust and the old man liDed to sit late because he was deaf and now at night it was quiet and he felt the difference.

From the word eCceXt we guess that x → C and p → X. From the word liDed we guess that k → D. Now we have decrypted the message:

it was very late and everyone had left the cafe except an old man who sat in the shadow the leaves of the tree made against the electric light. in the day time the street was dusty, but at night the dew settled the dust and the old man liked to sit late because he was deaf and now at night it was quiet and he felt the difference.

Exercise 1.8. Eve intercepts the following ciphertext (created by a substitution cipher):

DR DEV E OGBHSP BEUICR, EBH GV NRCDENV E SGUUSR JFRC-JNUGWGVUGX. EU UGWRV DR XEB ZR E SGUUSR VRSMGVD EBH GBXJBVGHRCEUR, ZIU DR DEV E YJJH DRECU EBH ESQEPV WREBV QRSS. UDR QEP DR HCRVVRV EBH DGV SJFR MJC XDRRVR GV ZEVRH JB EB RXXRBUCGX VXDJJS UREXDRC.

Decrypt this message.

Exercise 1.9. Eve intercepts the following ciphertext (created by a substitution cipher):

```
NPHB YXGMGYA LOSGODO MWPM WGA AGSOHYO NPJOA WGN MWO FOXIOYM AMXPGQWM NPH VGMW P
FPHMENGNO OTFXOAAGDOHOAA MWPM RXOV IPDECXPLSO YENFPXGAEHA ME LCAMOX JOPMEH. WO
REOA PM MGNOA NPJO REQ-SGJO HEGAOA, ACYW PA BOSFA PHR QXEVSGHQ.
```

Decrypt this message.

Exercise 1.10. Eve intercepts the following ciphertext (created by a substitution cipher):

```
ZA AXOUQG AWHRXJ "VUDXPMWVAG" WXF DAWFRXJ YWXQ NUUVG, RXTMSFRXJ HZA DAISNMRT,
NQ IMSHU (W XUF HU HZA FRGXAQ TZWDWTHAD UP HZA GWYA XWYA WXF W ISX UX IMWHU);
TDRYA WXF ISXRGZYAXH, NQ PRFU FUJGHUQAKGVQ (W ISX UX PQUFUD FUGHUQAKGVQ); WXF W
"ZUC-HU" JSRFA AXHRHMAF, AMATHDUXRTG PUD FUJG.
```

Decrypt this message.

Exercise 1.11. Eve intercepts the following ciphertext (created by a substitution cipher):

```
CBKGS RGKCDMGR TKUSWHUCSRDXN, TGK CXR ZMCTKDFDXG PURGM FUXTKSQFKDUX, KJG
PUIDG'T TJUK UXG BSCPG CK C KDPG, PUIDXN KJG PURGMT UB KJG FJCSCFKGST TMDNJKMW
KU NDIG KJG DPZSGTTDUX UB PUIGPGXK DX KJG BDXCM BDMP.
```

Decrypt this message.

Exercise 1.12. Eve intercepts the following ciphertext (created by a substitution cipher):

```
YOACERO KJ GTOZP OQFOCPZQD (ZJ IEZPLH) VOPRKQCBZGZOR CQF MZFORVPOCF VKVEBCPZGH,
GTO ATCPCAGOPR TCSO YOOQ FORAPZYOF CR VKRZGZSO ZQGOPQCGZKQCB ZAKQR KJ YKGT
WKFOPQ YPZGZRT AEBGEPO ZQ VCPGZAEBCP CQF GTO YPZGZRT VOKVBO ZQ DOQOPCB.
```

Decrypt this message.

There are many ways to modify the substitution cipher to make it less susceptible to frequency analysis. Perhaps the simplest way to strengthen the substitution cipher is to remove the spaces and punctuation between words. You can read about other modifications in p. 26-32 of *The Code Book* by Singh.

As was mentioned earlier, one of the drawbacks of the substitution cipher is that the keys are larger and more complicated that those of the shift cipher. One way around this problem is to

use a *keyword*. Suppose Alice and Bob agree that their keyword is `SECRET`. Using this keyword, Alice and Bob can from the following substitution key:

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
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
S	E	C	R	T	U	V	W	X	Y	Z	A	B	D	F	G	H	I	J	K	L	M	N	O	P	Q

Notice that `a` $\rightarrow$ `S` since `S` is the first letter of `SECRET`, and similarly `b` $\rightarrow$ `E`, `c` $\rightarrow$ `C`, and `d` $\rightarrow$ `R`. One might suspect that we'd then have `e` $\rightarrow$ `E`, but remember we have already assigned `b` $\rightarrow$ `E`. Instead, we assign `e` to the next letter in our keyword, which is `T`. Once the last letter of the keyword has been assigned, we then assign the rest of the letters with respect to alphabetical order, being careful to not repeat any letters that have already been assigned. The use of keywords does make breaking the substitution cipher easier, but the advantage is that there are still many possible keys and each of these keys is easy to remember.

2 The Vigenère Cipher and Friedman Attack

The *Vigenère cipher* was developed to thwart the frequency analysis attack to which the substitution cipher was susceptible. For many centuries, the Vigenére cipher was considered unbreakable. In 1925, William Friedman developed a method of breaking the Vigenère cipher and other similar *polyalphabetic substitution ciphers*. We will discuss how to encrypt and decrypt using the Vigenère cipher, and tomorrow discuss how this cipher can be broken.

To begin, Alice and Bob agree upon a key. For the Vigenère cipher, the key will consist of an m-tuple of integers $(a_1, \ldots, a_m)$ where m is some positive integer. To encrypt a message, Alice shifts the first letter of her plaintext a_1 places. She then shifts the second letter of her plaintext a_2 places. She continues in this way until she gets to the m-th letter, which she shifts by a_m. When Alice arrives at the $(m+1)$-th letter of her plaintext, she shifts that letter by a_1. She continues cycling through the different shift amounts as she encrypts her message.

As an example, suppose Alice wishes to send Bob the following message:

```
it's the wrong trousers gromit, and they've gone wrong!
```

Prior to sending the message, Alice and Bob decide to use the Vigenère cipher with a key of $(3, 7, 1)$. When Alice encrypts this message she will obtain the following ciphertext:

```
LA'T WOF ZYPQN UUVVTLSV NSRTJW, HOG AIHF'WH NPQL XUVOJ!
```

To decrypt a message, Bob simply shifts the first letter back a_1 places, the second letter back a_2 places and so on.

Exercise 2.1. Alice wishes to encrypt the following message using the Vigenère cipher with a key of $(6, 17, 2, 13)$:

```
i do like a bit of gorgonzola
```

Encrypt this message.

Exercise 2.2. Bob receives the following ciphertext from Alice:

```
OKGZULGQVISHGIWOGKWZMRPAAKHTKSIENMYW
```

The message was encrypted using the Vigenère cipher with a key of $(6, 17, 14, 12)$. Decrypt this message.

Suppose Eve intercepts the following ciphertext:

```
KUMLYITMKQUMIMLAYZXRPLLTGWTXHRWSIDIEUUTMJLIGHPVWDYOHTEXDLZXJKYBAIIVVML
KMINKQZBPIWSTZSPYQGKSVZFJNVULSILSBXPLVLEJKPTHPVZNFLVVWWAEBKEYAWGSQYIKQ
OHVWPANAGMCOBTRZTWKREUOTRZTGYEPOMKVKSTXHKCMKMJHKTRKLEBXDHTELEZKESPOMLS
JICMSQAABHAVNMLWABAIRHKTXEVVPEOHANGYLALEJKNKSIAPXRKUVHRAVNNWACMKXDVCZL
PAPXVADILEJFXEEYLQGXDLEHVHKTBOAAPTXHHSXMJTIBRADMKIPBZGIZZCFQAYIYXAYANQ
ILZTPSHGLSJHCZYOANBVOANHVKUMFSJAPBLWCMLMJJMUIYVUXEOHTMAWAMKQWUJNXOVUXX
ETMLMJZCFQAYBAINLIKIZHGLADLVMLAYMLXHLALRAZAHJPOMMMZLATRZAPXJAHZYYHJWEH
KMBAIOLIPEPLZTRZAPXMJJMLWWUBPMJKEAMYOJESSZIVVKZAMLAHNMINUWHRWULBRPVBAI
ACMGMJNUTOATMPMOONHVPOMIPWJQWMPFWYEHHSXMJAPXAKVLLY
```

She suspects this message has been encrypted using the Vigenère cipher. Is it possible for Eve to decrypt this message even if she doesn't know the key? If the message is long enough (as this one is), then it is possible for Eve to "break" this cipher. The attack we will use is called the *Friedman attack*.

The first step of the Friedman attack is to determine the key length. To do this we replicate the ciphertext and align the two ciphertext together, except that one copy is displaced by one character. We then highlights the characters which agree in the two ciphertext at the same point. We call such agreements, *coincidences*. Below, we highlight the coinscidences which occur when we consider a displacement of length 1:

```
 KUMLYITMKQUMIMLAYZXRPLLTGWTXHRWSIDIEUUTMJLIGHPVWDYOHTEXDLZXJKYBAIIVVMLKMINKQZBPIW
KUMLYITMKQUMIMLAYZXRPLLTGWTXHRWSIDIEUUTMJLIGHPVWDYOHTEXDLZXJKYBAIIVVMLKMINKQZBPIWS

STZSPYQGKSVZFJNVULSILSBXPLVLEJKPTHPVZNFLVVWWAEBKEYAWGSQYIKQOHVWPANAGMCOBTRZTWKREUO
TZSPYQGKSVZFJNVULSILSBXPLVLEJKPTHPVZNFLVVWWAEBKEYAWGSQYIKQOHVWPANAGMCOBTRZTWKREUOT

TRZTGYEPOMKVKSTXHKCMKMJHKTRKLEBXDHTELEZKESPOMLSJICMSQAABHAVNMLWABAIRHKTXEVVPEOHANG
RZTGYEPOMKVKSTXHKCMKMJHKTRKLEBXDHTELEZKESPOMLSJICMSQAABHAVNMLWABAIRHKTXEVVPEOHANGY

YLALEJKNKSIAPXRKUVHRAVNNWACMKXDVCZLPAPXVADILEJFXEEYLQGXDLEHVHKTBOAAPTXHHSXMJTIBRAD
LALEJKNKSIAPXRKUVHRAVNNWACMKXDVCZLPAPXVADILEJFXEEYLQGXDLEHVHKTBOAAPTXHHSXMJTIBRADM

MKIPBZGIZZCFQAYIYXAYANQILZTPSHGLSJHCZYOANBVOANHVKUMFSJAPBLWCMLMJJMUIYVUXEOHTMAWAMK
KIPBZGIZZCFQAYIYXAYANQILZTPSHGLSJHCZYOANBVOANHVKUMFSJAPBLWCMLMJJMUIYVUXEOHTMAWAMKQ

QWUJNXOVUXXETMLMJZCFQAYBAINLIKIZHGLADLVMLQYMLXHLALRAZAHJPOMMMZLATRZAPXJAHZYYHJWEHK
WUJNXOVUXXETMLMJZCFQAYBAINLIKIZHGLADLVMLAYMLXHLALRAZAHJPOMMMZLATRZAPXJAHZYYHJWEHKM

MBAIOLIPEPLZTRZAPXMJJMLWWUBPMJKEAMYOJESSZIVVKZAMLAHNMINUWHRWULBRPVBAIACMGMJNUTOATM
BAIOLIPEPLZTRZAPXMJJMLWWUBPMJKEAMYOJESSZIVVKZAMLAHNMINUWHRWULBRPVBAIACMGMJNUTOATMP

PMOONHVPOMIPWJQWMPFWYEHHSXMJAPXAKVLLY
MOONHVPOMIPWJQWMPFWYEHHSXMJAPXAKVLLY
```

We have discovered that there are 23 coincidences when we displace by 1 character. Next, we count the number of coincidences when we displace by 2 characters.

```
  KUMLYITMKQUMIMLAYZXRPLLTGWTXHRWSIDIEUUTMJLIGHPVWDYOHTEXDLZXJKYBAIIVVMLKMINKQZBPI
KUMLYITMKQUMIMLAYZXRPLLTGWTXHRWSIDIEUUTMJLIGHPVWDYOHTEXDLZXJKYBAIIVVMLKMINKQZBPIWS

WSTZSPYQGKSVZFJNVULSILSBXPLVLEJKPTHPVZNFLVVWWAEBKEYAWGSQYIKQOHVWPANAGMCOBTRZTWKREU
TZSPYQGKSVZFJNVULSILSBXPLVLEJKPTHPVZNFLVVWWAEBKEYAWGSQYIKQOHVWPANAGMCOBTRZTWKREUOT

OTRZTGYEPOMKVKSTXHKCMKMJHKTRKLEBXDHTELEZKESPOMLSJICMSQAABHAVNMLWABAIRHKTXEVVPEOHAN
RZTGYEPOMKVKSTXHKCMKMJHKTRKLEBXDHTELEZKESPOMLSJICMSQAABHAVNMLWABAIRHKTXEVVPEOHANGY

GYLALEJKNKSIAPXRKUVHRAVNNWACMKXDVCZLPAPXVADILEJFXEEYLQGXDLEHVHKTBOAAPTXHHSXMJTIBRA
LALEJKNKSIAPXRKUVHRAVNNWACMKXDVCZLPAPXVADILEJFXEEYLQGXDLEHVHKTBOAAPTXHHSXMJTIBRADM

DMKIPBZGIZZCFQAYIYXAYANQILZTPSHGLSJHCZYOANBVOANHVKUMFSJAPBLWCMLMJJMUIYVUXEOHTMAWAM
KIPBZGIZZCFQAYIYXAYANQILZTPSHGLSJHCZYOANBVOANHVKUMFSJAPBLWCMLMJJMUIYVUXEOHTMAWAMKQ

KQWUJNXOVUXXETMLMJZCFQAYBAINLIKIZHGLADLVMLQYMLXHLALRAZAHJPOMMMZLATRZAPXJAHZYYHJWEH
WUJNXOVUXXETMLMJZCFQAYBAINLIKIZHGLADLVMLAYMLXHLALRAZAHJPOMMMZLATRZAPXJAHZYYHJWEHKM

KMBAIOLIPEPLZTRZAPXMJJMLWWUBPMJKEAMYOJESSZIVVKZAMLAHNMINUWHRWULBRPVBAIACMGMJNUTOAT
BAIOLIPEPLZTRZAPXMJJMLWWUBPMJKEAMYOJESSZIVVKZAMLAHNMINUWHRWULBRPVBAIACMGMJNUTOATMP

MPMOONHVPOMIPWJQWMPFWYEHHSXMJAPXAKVLLY
MOONHVPOMIPWJQWMPFWYEHHSXMJAPXAKVLLY
```

We have discovered that there are 24 coincidences when we displace by 2 characters. Next, we count the number of coincidences when we displace by 3 characters.

```
  KUMLYITMKQUMIMLAYZXRPLLTGWTXHRWSIDIEUUTMJLIGHPVWDYOHTEXDLZXJKYBAIIVVMLKMINKQZBP
KUMLYITMKQUMIMLAYZXRPLLTGWTXHRWSIDIEUUTMJLIGHPVWDYOHTEXDLZXJKYBAIIVVMLKMINKQZBPIWS

IWSTZSPYQGKSVZFJNVULSILSBXPLVLEJKPTHPVZNFLVVWWAEBKEYAWGSQYIKQOHVWPANAGMCOBTRZTWKRE
TZSPYQGKSVZFJNVULSILSBXPLVLEJKPTHPVZNFLVVWWAEBKEYAWGSQYIKQOHVWPANAGMCOBTRZTWKREUOT
```

UOTRZTGYEPOMKVKSTXHKCMKMJHKTRKLEBXDHTELEZKESPOMLSJICMSQAABHAVNMLWABAIRHKTXEVVPEOHA
RZTGYEPOMKVKSTXHKCMKMJHKTRKLEBXDHTELEZKESPOMLSJICMSQAABHAVNMLWABAIRHKTXEVVPEOHANGY

NGYLALEJKNKSIAPXRKUVHRAVNNWACMKXDVCZLPAPXVADILEJFXEEYLQGXDLEHVHKTBOAAPTXHHSXMJTIBR
LALEJKNKSIAPXRKUVHRAVNNWACMKXDVCZLPAPXVADILEJFXEEYLQGXDLEHVHKTBOAAPTXHHSXMJTIBRADM

ADMKIPBZGIZZCFQAYIYXAYANQILZTPSHGLSJHCZYOANBVOANHVKUMFSJAPBLWCMLMJJMUIYVUXEOHTMAWA
KIPBZGIZZCFQAYIYXAYANQILZTPSHGLSJHCZYOANBVOANHVKUMFSJAPBLWCMLMJJMUIYVUXEOHTMAWAMKQ

MKQWUJNXOVUXXETMLMJZCFQAYBAINLIKIZHGLADLVMLQYMLXHLALRAZAHJPOMMMZLATRZAPXJAHZYYHJWE
WUJNXOVUXXETMLMJZCFQAYBAINLIKIZHGLADLVMLAYMLXHLALRAZAHJPOMMMZLATRZAPXJAHZYYHJWEHKM

HKMBAIOLIPEPLZTRZAPXMJJMLWWUBPMJKEAMYOJESSZIVVKZAMLAHNMINUWHRWULBRPVBAIACMGMJNUTOA
BAIOLIPEPLZTRZAPXMJJMLWWUBPMJKEAMYOJESSZIVVKZAMLAHNMINUWHRWULBRPVBAIACMGMJNUTOATMP

TMPMOONHVPOMIPWJQWMPFWYEHHSXMJAPXAKVLLY
MOONHVPOMIPWJQWMPFWYEHHSXMJAPXAKVLLY

We have discovered that there are 29 coincidences when we displace by 3 characters. Next, we count the number of coincidences when we displace by 4 characters.

KUMLYITMKQUMIMLAYZXRPLLTGWTXHRWSIDIEUUTMJLIGHPVWDYOHTEXDLZXJKYBAIIVVMLKMINKQZB
KUMLYITMKQUMIMLAYZXRPLLTGWTXHRWSIDIEUUTMJLIGHPVWDYOHTEXDLZXJKYBAIIVVMLKMINKQZBPIWS

PIWSTZSPYQGKSVZFJNVULSILSBXPLVLEJKPTHPVZNFLVVWWAEBKEYAWGSQYIKQOHVWPANAGMCOBTRZTWKR
TZSPYQGKSVZFJNVULSILSBXPLVLEJKPTHPVZNFLVVWWAEBKEYAWGSQYIKQOHVWPANAGMCOBTRZTWKREUOT

EUOTRZTGYEPOMKVKSTXHKCMKMJHKTRKLEBXDHTELEZKESPOMLSJICMSQAABHAVNMLWABAIRHKTXEVVPEOH
RZTGYEPOMKVKSTXHKCMKMJHKTRKLEBXDHTELEZKESPOMLSJICMSQAABHAVNMLWABAIRHKTXEVVPEOHANGY

ANGYLALEJKNKSIAPXRKUVHRAVNNWACMKXDVCZLPAPXVADILEJFXEEYLQGXDLEHVHKTBOAAPTXHHSXMJTIB
LALEJKNKSIAPXRKUVHRAVNNWACMKXDVCZLPAPXVADILEJFXEEYLQGXDLEHVHKTBOAAPTXHHSXMJTIBRADM

RADMKIPBZGIZZCFQAYIYXAYANQILZTPSHGLSJHCZYOANBVOANHVKUMFSJAPBLWCMLMJJMUIYVUXEOHTMAW
KIPBZGIZZCFQAYIYXAYANQILZTPSHGLSJHCZYOANBVOANHVKUMFSJAPBLWCMLMJJMUIYVUXEOHTMAWAMKQ

AMKQWUJNXOVUXXETMLMJZCFQAYBAINLIKIZHGLADLVMLQYMLXHLALRAZAHJPOMMMZLATRZAPXJAHZYYHJW
WUJNXOVUXXETMLMJZCFQAYBAINLIKIZHGLADLVMLAYMLXHLALRAZAHJPOMMMZLATRZAPXJAHZYYHJWEHKM

EHKMBAIOLIPEPLZTRZAPXMJJMLWWUBPMJKEAMYOJESSZIVVKZAMLAHNMINUWHRWULBRPVBAIACMGMJNUTO
BAIOLIPEPLZTRZAPXMJJMLWWUBPMJKEAMYOJESSZIVVKZAMLAHNMINUWHRWULBRPVBAIACMGMJNUTOATMP

ATMPMOONHVPOMIPWJQWMPFWYEHHSXMJAPXAKVLLY
MOONHVPOMIPWJQWMPFWYEHHSXMJAPXAKVLLY

We have discovered that there are 24 coincidences when we displace by 4 characters. Next, we count the number of coincidences when we displace by 5 characters.

KUMLYITMKQUMIMLAYZXRPLLTGWTXHRWSIDIEUUTMJLIGHPVWDYOHTEXDLZXJKYBAIIVVMLKMINKQZ
KUMLYITMKQUMIMLAYZXRPLLTGWTXHRWSIDIEUUTMJLIGHPVWDYOHTEXDLZXJKYBAIIVVMLKMINKQZBPIWS

BPIWSTZSPYQGKSVZFJNVULSILSBXPLVLEJKPTHPVZNFLVVWWAEBKEYAWGSQYIKQOHVWPANAGMCOBTRZTWK
TZSPYQGKSVZFJNVULSILSBXPLVLEJKPTHPVZNFLVVWWAEBKEYAWGSQYIKQOHVWPANAGMCOBTRZTWKREUOT

REUOTRZTGYEPOMKVKSTXHKCMKMJHKTRKLEBXDHTELEZKESPOMLSJICMSQAABHAVNMLWABAIRHKTXEVVPEO
RZTGYEPOMKVKSTXHKCMKMJHKTRKLEBXDHTELEZKESPOMLSJICMSQAABHAVNMLWABAIRHKTXEVVPEOHANGY

HANGYLALEJKNKSIAPXRKUVHRAVNNWACMKXDVCZLPAPXVADILEJFXEEYLQGXDLEHVHKTBOAAPTXHHSXMJTI
LALEJKNKSIAPXRKUVHRAVNNWACMKXDVCZLPAPXVADILEJFXEEYLQGXDLEHVHKTBOAAPTXHHSXMJTIBRADM

BRADMKIPBZGIZZCFQAYIYXAYANQILZTPSHGLSJHCZYOANBVOANHVKUMFSJAPBLWCMLMJJMUIYVUXEOHTMA
KIPBZGIZZCFQAYIYXAYANQILZTPSHGLSJHCZYOANBVOANHVKUMFSJAPBLWCMLMJJMUIYVUXEOHTMAWAMKQ

WAMKQWUJNXOVUXXETMLMJZCFQAYBAINLIKIZHGLADLVMLQYMLXHLALRAZAHJPOMMMZLATRZAPXJAHZYYHJ
WUJNXOVUXXETMLMJZCFQAYBAINLIKIZHGLADLVMLAYMLXHLALRAZAHJPOMMMZLATRZAPXJAHZYYHJWEHKM

WEHKMBAIOLIPEPLZTRZAPXMJJMLWWUBPMJKEAMYOJESSZIVVKZAMLAHNMINUWHRWULBRPVBAIACMGMJNUT
BAIOLIPEPLZTRZAPXMJJMLWWUBPMJKEAMYOJESSZIVVKZAMLAHNMINUWHRWULBRPVBAIACMGMJNUTOATMP

OATMPMOONHVPOMIPWJQWMPFWYEHHSXMJAPXAKVLLY
MOONHVPOMIPWJQWMPFWYEHHSXMJAPXAKVLLY

We have discovered that there are 38 coincidences when we displace by 5. Next, we count the number of coincidences when we displace by 6 characters.

KUMLYITMKQUMIMLAYZXRPLLTGWTXHRWSIDIEUUTMJLIGHPVWDYOHTEXDLZXJKYBAIIVVMLKMINKQ
KUMLYITMKQUMIMLAYZXRPLLTGWTXHRWSIDIEUUTMJLIGHPVWDYOHTEXDLZXJKYBAIIVVMLKMINKQZBPIWS

ZBPIWSTZSPYQGKSVZFJNVULSILSBXPLVLEJKPTHPVZNFLVVWWAEBKEYAWGSQYIKQOHVWPANAGMCOBTRZTW
TZSPYQGKSVZFJNVULSILSBXPLVLEJKPTHPVZNFLVVWWAEBKEYAWGSQYIKQOHVWPANAGMCOBTRZTWKREUOT

KREUOTRZTGYEPOMKVKSTXHKCMKMJHKTRKLEBXDHTELEZKESPOMLSJICMSQAABHAVNMLWABAIRHKTXEVVPE
RZTGYEPOMKVKSTXHKCMKMJHKTRKLEBXDHTELEZKESPOMLSJICMSQAABHAVNMLWABAIRHKTXEVVPEOHANGY

OHANGYLALEJKNKSIAPXRKUVHRAVNNWACMKXDVCZLPAPXVADILEJFXEEYLQGXDLEHVHKTBOAAPTXHHSXMJT
LALEJKNKSIAPXRKUVHRAVNNWACMKXDVCZLPAPXVADILEJFXEEYLQGXDLEHVHKTBOAAPTXHHSXMJTIBRADM

IBRADMKIPBZGIZZCFQAYIYXAYANQILZTPSHGLSJHCZYOANBVOANHVKUMFSJAPBLWCMLMJJMUIYVUXEOHTM
KIPBZGIZZCFQAYIYXAYANQILZTPSHGLSJHCZYOANBVOANHVKUMFSJAPBLWCMLMJJMUIYVUXEOHTMAWAMKQ

AWAMKQWUJNXOVUXXETMLMJZCFQAYBAINLIKIZHGLADLVMLQYMLXHLALRAZAHJPOMMMZLATRZAPXJAHZYYH
WUJNXOVUXXETMLMJZCFQAYBAINLIKIZHGLADLVMLAYMLXHLALRAZAHJPOMMMZLATRZAPXJAHZYYHJWEHKM

JWEHKMBAIOLIPEPLZTRZAPXMJJMLWWUBPMJKEAMYOJESSZIVVKZAMLAHNMINUWHRWULBRPVBAIACMGMJNU
BAIOLIPEPLZTRZAPXMJJMLWWUBPMJKEAMYOJESSZIVVKZAMLAHNMINUWHRWULBRPVBAIACMGMJNUTOATMP

TOATMPMOONHVPOMIPWJQWMPFWYEHHSXMJAPXAKVLL
MOONHVPOMIPWJQWMPFWYEHHSXMJAPXAKVLLY

We have discovered that there are 15 coincidences when we displace by 6 characters. We record our observations in the following table.

Displacements	1	2	3	4	5	6
Coincidences	23	24	29	24	38	15

The basic idea behind the Friedman attack is this: the length of the key is *probably* equal to the displacement with the most coincidences. If we suspect that Alice encrypted using a key of at most length 6, which is what we always assume for our purposes, then it follows that we need only count the number of coinscidences for displacements of 1 to 6.

From our calculations, it appears that the message was encrypted using a key of length 5. In light of this, we take our ciphertext and break it into 5 different pieces. The first piece will consist of the first letter, sixth letter, eleventh letter, ... of the ciphertext; the second piece will consist of the second letter, seventh letter, twelfth letter, ... of the ciphertext; and so on. This will leave us with the following pieces:

Piece #1:

KIUAPWWEJPODKIKQWPSNIPJPLAYQOACZEZPKKJKDEPJQAWREOYJIKAADPAJYD
HAHJAPZAAISJOOKJWJYOWWOEJANZDAHAPZZAHKOPZJWJYSKANWPAJAOPWPHJK

Piece #2:

UTMYLTSULVHLYVMZSYVVLLKVVEAYHNOTUTOSCHLHZOIAVAHVHLKAUVCVADFLL
KAHTDBZYYLHHAAUACJVHAUVTZYLHLYLZOLAHJMLLAJUKOZZHUUVCNTOOJFHAV

Piece #3:

MMIZLXIUIWTZBVIBTQZUSVPZVBWIVABWOGMTMKETKMCANBKVAANPVNMCPIXQE
TPSIMZCIAZGCNNMPMMUTMJUMCBIGVMAAMAPZWBIZPMBEJIANWLBMUMNMQWSPL

Piece #4:

LKMXTHDTGDEXAMNPZGFLBLTNWKGKWGTKTYKXKTBEELMBMATPNLKXHNKZXLEGH
BTXBKGFYNTLZBHFBLUXMKNXLFAKLMLLHMTXYEAPTXLPAEVMMHBAGTPHIWYXXL

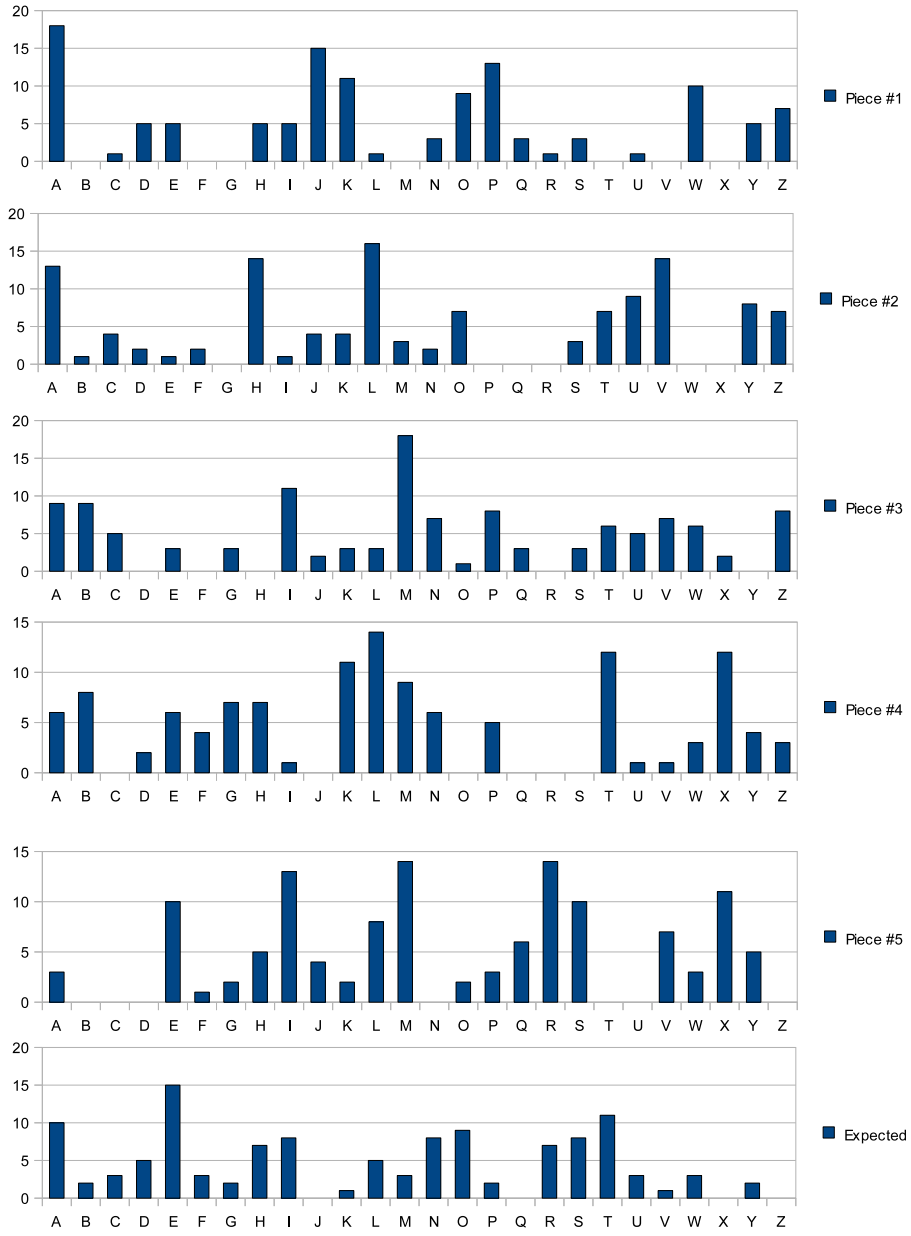
Piece #5:

YQLRGRIMHYXJILKISKJSXEHFWESQPMRRREVHMRXLSSSHLIXEGESRRWXLVEEXV
OXMRIIQXQPSYVVSLMIEAQXXMQIIALXRJMRJYHIERMWMMSVLIRRIMOMVPMEMAY

For each piece, we count the number of times each character occurs. We record our results in the table below. Also included in the table is a column that tells us the expected number of times a character should occur provided there was no shift. To compute this column, we took the frequency of each letter as it normally occurs in the English language (see previous section for table) and multiplied it by the total number of characters in each piece (122) to arrive at the expected number of times each character should appear in a text of 122 characters.

Characters	Piece #1	Piece #2	Piece #3	Piece #4	Piece #5	Expected
A	18	13	9	6	3	10
B	0	1	9	8	0	2
C	1	4	5	0	0	3
D	5	2	0	2	0	5
E	5	1	3	6	10	15
F	0	2	0	4	1	3
G	0	0	3	7	2	2
H	5	14	0	7	5	7
I	5	1	11	1	13	8
J	15	4	2	0	4	0
K	11	4	3	11	2	1
L	1	16	3	14	8	5
M	0	3	18	9	14	3
N	3	2	7	6	0	8
O	9	7	1	0	2	9
P	13	0	8	5	3	2
Q	3	0	3	0	6	0
R	1	0	0	0	14	7
S	3	3	3	0	10	8
T	0	7	6	12	0	11
U	1	9	5	1	0	3
V	0	14	7	1	7	1
W	10	0	6	3	3	3
X	0	0	2	12	11	0
Y	5	8	0	4	5	2
Z	7	7	8	3	0	0

The following images show the distributions of letters for each piece graphically.



In our table we should expect each column to roughly agree with the Expected column up to some vertical shift. By determining which shifts match up best, we suspect that the key for this cipher was $(22, 7, 8, 19, 4)$. We can perform the same computation in a graphical way, by shifting the graph for a given piece so that it roughly matches up with the "Expected" graph. When we decrypt using this key our message becomes the following (after adding some spaces

and punctuation):

```
one summer, my father rented a camp on a lake in maine and took us all
there for the month of august.  we all got ringworm from some kittens
and had to rub ponds extract on our arms and legs night and morning,
and my father rolled over in a canoe with all his clothes on; but
outside of that the vacation was a success and from then on none of us
ever thought there was any place in the world like that lake in maine.
we returned summer after summer--always on august first for one month.
i have since become a salt-water man, but sometimes in summer there are
days when the restlessness of the tides and the fearful cold of the sea
water and the incessant wind which blows across the afternoon and into
the evening make me wish for the placidity of a lake in the woods.
```

Notice that our plaintext, and consequently the ciphertext, were very large in comparison to most of our earlier examples. We chose long examples because we required many characters in order to perform frequency analysis on each of the five pieces we obtained.

Exercise 2.3. The following ciphertext was produced using the Vigenère cipher:

```
EASNZHFGLLCZPGSNMROIZNBQXTBRTLTKNXODQBFCEPCBPMVOXXZKYVVYWRSHAKSCDBCXLEAYD
MRODICXOXBMJHTYYXKRZMFKGXZCLPWVOTBNMESKVKCKOTHXTZVDQTZVLGRKWHBOMNHCZHBLCB
UREXBOONDGSXBRPLOGEASUTGRVJPOBXMVYQAWCCXQOAMWYYASPPEHRTLVOLKHCAKWXRYCBHTF
NEHAOPMHRPFOVWYFYXMVOZERGZFOXHACGTISNLVVKTKKSEAVOCTDBZGHYEASVTMHVPVVSWWHR
LMVOWWCEEBHCLKACEHVSXHBOREOXNXOXOLASWXDVLVSNEASCEKOXRXFYYTTYZMWXRHTSYGCMP
GHPLFWVTTFSERKSEAHRPXZNPLHNLNUREXF
```

Decrypt this message.

3 The Enigma Machine

As we can we have already seen, a major weakness with the Vigenère cipher is that the shift cipher is used, albeit on separate partitions of the plaintext. One means of improving the Vigenère cipher is to consider m separate substitution cipher permutations: σ_1, σ_2, ..., σ_m and then encrypt a plaintext message by using σ_1 to encrypt the first letter, σ_2 to encrypt the second letter, ..., σ_m to encrypt the m-th letter and then back to σ_1 to encrypt the $(m+1)$-th letter. One continues encrypting letters by following this pattern.

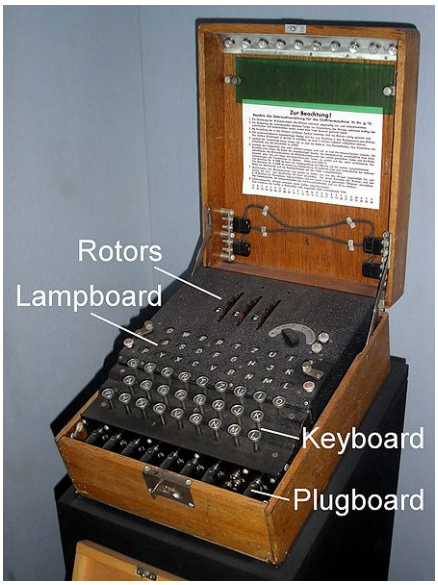
One of the obvious problems with this encryption technique is that the key could be quite large, since it would consist for m distinct permutations. If m is small then we have the problem that this method of encryption might still be broken by modifying the Friedman Attack (although breaking this cipher would be much harder than breaking the Vigenére cipher). One way to make a Friedman-like attack more difficult would be to make m large. But that brings us back to the problem of having a very large key.

The German inventor Arthur Scherbius recognized that, although this cipher is impractical to perform by hand, it may be easily implemented as a machine. His invention, which was later to be become known as the Enigma machine, was an electro-mechanical device that more or less performed this encryption algorithm. The Enigma machine consisted of a keyboard, letter display, and complicated circuitry which encrypted (or decrypted) typed in letters. It was later the primary means of encryption for Nazi Germany during World War II.

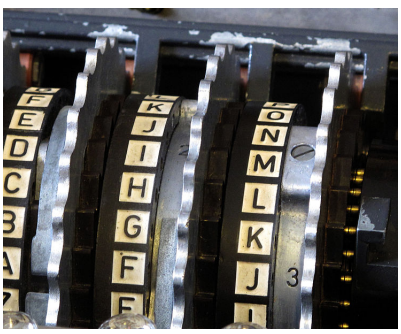
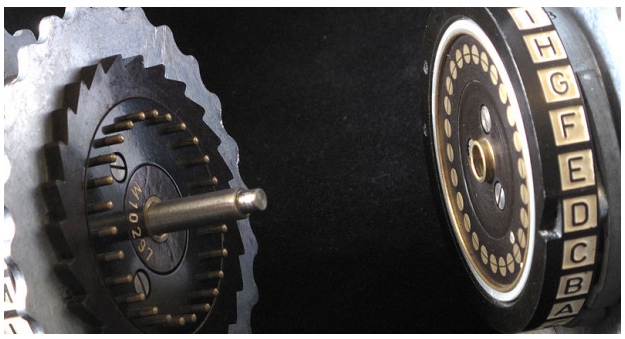
The circuitry for the Enigma machine was placed on three rotors, which would rotate once a letter had been typed. The rotation of the rotors would "create" a new permutation to be used for the next letter. Thus a different permutation would be used each time a letter was typed. In all, an Enigma user would have to type in approximately 105456 characters before a permutation would be repeated. In effect, Scherbius devised a machine that could perform our aforementioned cipher with $m = 105456$.

One of the design "features" of the Enigma machine was that the decryption process was almost identical to the encryption process. When Alice would encrypt a message using an Enigma machine, she could then send the message to Bob, and he would then type in the ciphertext and the Enigma machine would then give back the plaintext. The only hitch was that Alice and Bob had to make sure they both started their encryption/decryption process with the exact same permutation. This choice of initial permutation was essentially the key and as we already mentioned, this meant that Alice and Bob would have 105456 possible keys to choose from.[1] Since the Enigma machine was mechanical in nature, there was some "work" involved in configuring the machine to work with a given key. Next we describe the various parts of the Enigma machine and how a given key for the Enigma machine would determine the arrangements of these parts.

[1]We are currently dealing with a simplified version of the Enigma machine. The real Enigma machine had many more possible keys, but as we will explain later, these many extra keys didn't make Enigma machine ciphertext much more secure than if there were only 105456.



Each rotor was labeled by a number, either: 1,2, or 3. Before encrypting a message, the user of the Enigma machine would have to load the rotors into the machine in a specific order. For instance, a user could load rotor 1 first, then rotor 3, and then rotor 2; or a user could load rotor 3 first, followed by rotor 1, and then rotor 2. In all, there were 6 different ways in which the rotors could be loaded. Furthermore, once a rotor had been loaded, it could be rotated to one of 26 possible initial position. These initial positions were labeled by letters of the alphabet. Thus an Enigma user could for instance take rotor 1 and load it to position `Q`, rotor 3 and load it to position `A`, and rotor 2 and load it to position `E`. It was by specifying a rotor arrangement, such as 2-3-1 combined with initial positions for the rotors, such as Q-T-A, that one was able to specify a key for the Enigma machine.[2]

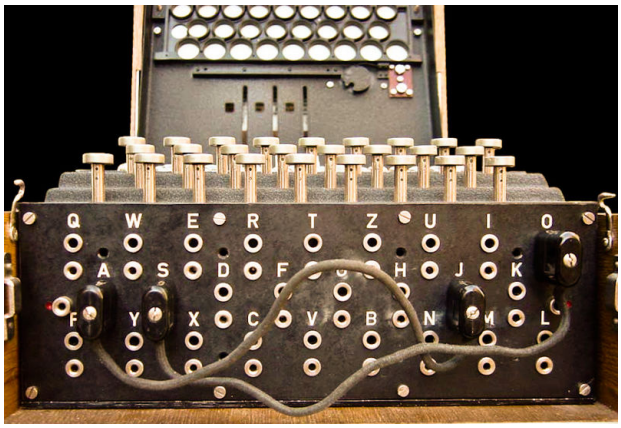
Although 105456 may seem like a large number of keys, Scherbius recognized that these were not enough keys to thwart a dedicated opponent. To add an extra layer of complexity to

[2]Since there are 26 letters in the alphabet and 6 ways to arrange the rotors we see that there are roughly $6 \cdot 26^3 = 105456$ possible choices for a key.

Enigma, there was also a plug board. By connecting wires on the plug board, an Enigma user could "swap" any two letters. The Enigma machine came with six wires, so this allowed for six pairs of letters to be swapped in this way. There are $100,391,791,500$ ways to wire the plug board. Thus there are

$$105,456 \cdot 100,391,791,500 \approx 10,000,000,000,000,000$$

possible keys (even when we have only three rotors!). Although the Enigma machine clearly had a large number of keys, our experience with the substitution cipher (which had even more possible keys) should tell us that having a large number of keys does not necessarily make a cipher strong. As we will see, the allied forces were able to exploit some less than obvious weaknesses in the Enigma machine to decode many of Germany's messages.



One of the tricks to the Allied forces breaking the Enigma machine was actually not so much the fault of the machine itself, but in how it was used. To make sure that all Enigma machine users were working with the same key the German military dispersed a *code book* to each user. In this book, they would be told which key to use for a given day. We will refer to this key as the *day key*. The keys given in the code book would tell the user how to configure all these different parts of the Enigma machine. A typical key entry would be something like the following:

Rotor Arrangement:	2-3-1.
Rotor Initial Position:	E-Y-B.
Plug board Settings:	B/D-T/A-S/Y-Q/V-X/H-K/O.

The Germans feared that if enough messages were sent using the day key, then the allies might be able to decode their messages. To help prevent this, they decided that each message would also have its own individual *message key*. A message key was much simpler than a day key; it simply consisted of three letters which indicated a different initial rotor position than that of the day key. Here is how this would work:

An Enigma operator might decide to use a message key of A-G-X. The operator would then configure his machine to use the day key, say for instance, the day key we used as an example

earlier, and then he would encrypt the message key: AGX. To make sure that there wasn't any mis communications, the operator would usually type the message key twice to make sure that the message key was properly transmitted. After doing this, the Enigma operator would then change the initial positions of his rotors to that of the message key: A-G-X. He would then proceed with typing out the rest of the message.

The Polish cipher bureau was interested in decrypting Germany's messages and was able, through the use of a spy, to determine how the Enigma machine was constructed and how it was used by the Germans. In particular, they knew about how the message keys were transmitted. This left the bureau with the task of determining the day key from the ciphertexts they intercepted. Although this might seem like an impossible task, Marian Rejewski, an employee of the cipher bureau, figured out how this could be done.

Suppose the cipher bureau intercepted the following ciphertexts on a given day:

```
AYIDUV...
WIONOU...
EZNWIP...
NTIENC...
```

We only look at the first six letters since these were the only letters encrypted using the day key. We know that these first six letters in each of these ciphertexts came from encrypting a message key. So for instance, if in the first message, the Enigma operator chose a message key of AGX then the first six letters he would type into his machine using the day key would be: `agxagx`. Thus the first and fourth letter of each ciphertext were both encrypting the same letter. Similarly, the second and fifth letter and the third and sixth letter were also encrypting identical letters. By intercepting many more such ciphertexts, the Poles were able to from tables such as the one below, where the top row gives the first letter of a ciphertext and the bottom row gives the corresponding fourth letter of the ciphertext.

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
D	K	T	C	W	O	J	L	A	U	P	G	B	E	R	I	V	F	M	S	X	Z	N	Y	Q	H

Obviously other such tables could be made from the second and fifth letters of the ciphertext and the third and sixth letters of the ciphertext. Rejewski thought of each table as defining a permutation. For example, the prior table would give the following permutation:

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
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
D	K	T	C	W	O	J	L	A	U	P	G	B	E	R	I	V	F	M	S	X	Z	N	Y	Q	H

By looking at this table Rejewski would form "chains". He would take a letter, see where the permutation sent it and then plug the resulting letter back into the permutation. By doing this he would obtain the following:

$$\mathtt{A} \to \mathtt{D} \to \mathtt{C} \to \mathtt{T} \to \mathtt{S} \to \mathtt{M} \to \mathtt{B} \to \mathtt{K} \to \mathtt{P} \to \mathtt{I} \to \mathtt{A}$$

$$\texttt{J} \to \texttt{U} \to \texttt{X} \to \texttt{Y} \to \texttt{Q} \to \texttt{V} \to \texttt{Z} \to \texttt{H} \to \texttt{L} \to \texttt{G} \to \texttt{J}$$

$$\texttt{E} \to \texttt{W} \to \texttt{N} \to \texttt{E}$$

$$\texttt{F} \to \texttt{O} \to \texttt{R} \to \texttt{F}$$

We would then say that this permutation gives a chain of lengths: 10, 10, 3, 3.

Rejewski's key insight was this: the choice of plug board settings doesn't determine the length of a given chain. Recall that there are only 105,391 ways to configure the rotors. Since the Polish cipher bureau had constructed a copy of the Enigma machine, they were able to reference rotor configurations according to these chain lengths. Thus when they intercepted enough ciphertexts, they were able to determine the rotor configuration being used based upon the chain lengths.

Recall that only six pairs of letters are swapped by the plug board. This means that there is approximately a probability of 1/8 that the message key will consist of letters that have not been swapped by the plug board. Thus, the Poles were able to find the message key for about 1/8 of their intercepted ciphertexts. At this point such a message could be deciphered, although with incorrect plug board settings. Fortunately, it was a simple matter to deduce the correct plug board settings from these slightly incorrect messages and thus determine the entirety of the day key.

The following problems will make use of the Enigma Applets I've included. To simplify matters we've decided that our Enigma machine has no plug board.

Exercise 3.1. The Polish Cipher Bureau has intercepted the following 26 messages:

Message #1: `ULJFPAWUAZSVIYBCJFVLA`
Message #2: `MDFRZPGDFFJLUSESVICMS`
Message #3: `WOHWBVBCTMGMVZSFYZOQS`
Message #4: `BWVZNBTFTPKHBYLQCXHTF`
Message #5: `TBTJLNAAPXHMXWDVQDFEN`
Message #6: `RVWEDLTNFPLQIXSPRWIWX`
Message #7: `PAIOXYKJVUMBSMMLVQKEI`
Message #8: `LUQTYMWIMNWJRHIKROXXQ`
Message #9: `VZDUEXSJOSKNIQNOVXPRX`
Message #10: `GGEKSGITUUWHHVMJAAJHJ`
Message #11: `NRBLJCXEFCAMEZQZVQWQF`
Message #12: `EIZSMSEETWUXPRVBSHFBI`
Message #13: `OEUHKOHNCOVXUJRSFGXJF`
Message #14: `XHNMQTXLGXTSADXGMEGDY`
Message #15: `CMSPWZNXXLPKTUCCYHJKP`
Message #16: `SQGBADZDNDIRVJQZQKMOY`
Message #17: `KYXVFRYLGHFWIGVTTQAWN`
Message #18: `ZCRIRWVLDBFSUVMVYKYIC`
Message #19: `APYGOUZJUTXIPWBNGPWDI`
Message #20: `JKANVEECDIMGFKBCYTKKI`
Message #21: `YNCAIIJQSZWJBDTTNIJIX`
Message #22: `IJPXTJZBHZHTICZRAEFDM`
Message #23: `QTMQCKQQUBVJJIMSMNCUI`
Message #24: `FSLDUFMOQEVDNIBCKMLOP`
Message #25: `HXKCHHDSRRCJCQUXZMJYQ`
Message #26: `DFOYGQPEADQZQFITZOP`

Determine what the day key is and then decrypt the remaining parts of ciphertext by using the message keys.

Exercise 3.2. The Polish Cipher Bureau has intercepted the following 26 messages:

Message #1: TQRMHVSFBBAFROYOCHIIGMJWTTSWXWVB
Message #2: CVNBBHBLMAVTYUTIURQOTNMNXGPINFDU
Message #3: QWPDNKHOMFGXBYBFNFNKFXNIBAVHTHJT
Message #4: RBTWJERRIDDQDVYMDKLJWVISVDVNZNLO
Message #5: ORWSCJOZXYWRVIVCRJSWYLAEIZNYXFNS
Message #6: MMARKQFFBCMSKUEDQUFKJAGIZEZSASEH
Message #7: GEJZFFZHFPFTBREGAXEMLFORKGVGXMBC
Message #8: FHXTZYUHCENHUZRFYEYLSKRPQOEFJJST
Message #9: ZPYYVUJJPJDHJGYKXTZHZMKIPVAGTUKK
Message #10: DZZCOITQEDFKYKBZJSZEPHLWXSKIRQAH
Message #11: UXCVDZJEDRSYGVSMTODGLQKUZDUCHGGZ
Message #12: EGQJSDGBWXSCRMIQWPSYLAIRPPCZIOUI
Message #13: BUIQMXNYDYUJNQITFJPFMANXVNLUDSZI
Message #14: SIBEQRKAGUANYQQSWUDNEYYXGISZWKRT
Message #15: VFGILORVLQPGGMPNWRDDZHCSDYDMYUEU
Message #16: NDOFGCPIXYISFJBKEAASRAMLRURHBWCY
Message #17: XYSATLJDUYKMHPPRYXSYHLBHXNXBVJSZ
Message #18: JSKOPWRDCNZDOEAWQAXEMYEJZHFYNZJM
Message #19: PTFLIMVXHDNENOATCYUYXZUOGIHUZKYO
Message #20: KLVHENJTFGMMXPXOSTPWRGZGORUCNEDJ
Message #21: YCENRALRXVSVLCLARGLPPTVLDXTSMDIX
Message #22: HOMUASDNDXSUMQUKDGTRYBCOKEZKNSGE
Message #23: INLPWPFKXECOJZOVQFZFJEWJXAZRKNKF
Message #24: WKUGUGYYQDVYVFFMHCJZJFBTEIETIEZB
Message #25: LADXYTLINRRSLRZMJOJHLEJXYYIEJBDZ
Message #26: AJHKXBCHKHGAVFTBUWLYLAOPBKKHTFVZ

Determine what the day key is and then decrypt the remaining parts of ciphertext by using the message keys.

Exercise 3.3. The Polish Cipher Bureau has intercepted the following 26 messages:

Message #1: VHAJWROHSSYUIVUWLLQPNLDRBNHRVJFX
Message #2: TNQTICNUSOIPYQXVRUOJWWYLZCEVXLVW
Message #3: XEVXJTUVABZHANDPTLVGHVLUFQZLCAVA
Message #4: AGFAONWGNANENUQWUYGQXUQZPFYMNUCD
Message #5: BMUPPZDONTAFIBJYFGCEGYNLAUWMYNYD
Message #6: SOCENYDLRDCAPQKYDCMAXSNSZTTVNYAP
Message #7: WTNZAOEPNEYJCNXFBMDCCWWAWWMOPWDY
Message #8: ZSHRYIRFSIVUPFZBAIAOZTIOHBMYHSQX
Message #9: MJPIBVUZNNABWDVDFMFJDADXMSPBAPTW
Message #10: JCDWVLGZOSAPLDZRHDPTGXBIMNUPAXUY
Message #11: GBLMLBDJQGDMXFJTPIFFNZVXXEJNSDEW
Message #12: CKIOKKFYEQHIWOYGJHEFAITDQJAUSJSR
Message #13: HPSSSGNEIDRSMQIRNMSDCWGSTNPVXAZD
Message #14: PDRLDQNFVWOJHCKVKTIQTVRYDARADYLT
Message #15: IXYVXFCHXPHQNRBLTCXENXMBLYLBFIOX
Message #16: QZOQEXVBQINWFRFXALRPVRQLPJWEVOQH
Message #17: LIBCZMOURUHPUAZFXNKRZCXVNYQBWOFZ
Message #18: EAMUUWVSBFFFYRFERXIHBFLYWBDLBFUL
Message #19: DWTBQPHKHBUCQHWISIDNVUPITYRGFYQX
Message #20: FYJGTADEYEHRMJJJNZUUXXPTEQSELYHW
Message #21: UQWDMENWNCHLRQTFKOWTQULUGTOYYFYX
Message #22: RLEKCSMMOSNIZBDGYNDOZWAYYQWYUCFD
Message #23: NFGHHULJUOCVELBOPAJULSRZHJZKZSJR
Message #24: YVZFGDZTBEZASRDIIXAMORLZJVOCTYGT
Message #25: KUXYFJGPUFWJZIXBWPUFALINJDRLVNFB
Message #26: ORKNRHIQTFMYSGVHQUQDZIBKYXQOETGG

Determine what the day key is and then decrypt the remaining parts of ciphertext by using the message keys. *Hint: When using the bombe applet, only consider the first two keys given to you.*

Exercise 3.4. The Polish Cipher Bureau has intercepted the following 26 messages:

Message #1: QNHFOTBIQIMKHNKVTXSYDTIRTQVYSKZC
Message #2: HXFYXVXCKCDDFVTDJEOIDKUQBNLQCZRI
Message #3: LRAPWRLVYNPYOGWCYYIIYYRIBJKXXOSQ
Message #4: CLYDHOBBGWJSHRFBWXHGTFZPWQHZADVL
Message #5: KKDEKXBPRHAYQCAIKXCEXRSELIYBGNHJ
Message #6: FIPXAPKNZNWBDMRZODRQZNAOQXWBXPDN
Message #7: ZDRWQSQPDGUMNUVPTMJQRWDTLAAIVCYY
Message #8: RGBBZDMJEIXLXXTDZYIGFPQXRZWHCVTJ
Message #9: WHNMUIPKEIZMMAXAJLMVNKOIMDVEMZMP
Message #10: XWTCJETYTRDMILWBSHHWADGTGDSIVXRZ
Message #11: PUKOGHRQVHMOJKHXZWIMKSYUTYDDXFCP
Message #12: STLGSFCUZFZTGENNSTENBWBDYNMAEFGO
Message #13: IJEANLJCOKTZRQDOZQBFSPOYMCJZVFUQ
Message #14: UPSHPQAKAKVYZVNOKBYMZTZVNNRZAGYG
Message #15: JEMSEBCJYANPQWYTYRQWZLFVRYQRWSWE
Message #16: EQXQCKZFRGLAIDJQJHRQZZIVVIVPRICE
Message #17: MCGTLGYBZGAMOZCVDFQFVPTEYBUZHVHH
Message #18: BMIZRAOOSENYQZVQDDQECTVYARTQLCMG
Message #19: VOCRYNVYIFCZHGUHPHEQLMSXVKQUYWIV
Message #20: TAZIDJNZUAHORMBGEZOYJNMPXWNJOSSY
Message #21: DZUUIUWJOHHLBMGAOPLPKSAVJKIEWJDG
Message #22: OYOLTCGGHEQZBCKGEQCJWAPRPGQDOJDS
Message #23: NVJVMYZMLSCBHZDFXZTPEJFEHNFOLRXV
Message #24: ASWNVWJQDORJXRVVHUHGUADCXPGYDPIE
Message #25: YFVKFMFPNGWXYJOONNMIYXKRLIYARMOO
Message #26: GBQJBZZBGCWEMCUDCSOLIOSAMYSUTNIS

Determine what the day key is and then decrypt the remaining parts of ciphertext by using the message keys.

Exercise 3.5. The Polish Cipher Bureau has intercepted the following 26 messages:

Message #1: `RKUKURVPPZCTZAAFPYFKIDHXYGSTVEFX`
Message #2: `IJDUJOCMWTOOFXEGDZDMMENWZCXVVGCA`
Message #3: `TYHZYXNMEYIKHLVMTOOZSAAKMGBZBGCF`
Message #4: `QSLNCHXUSAGGZFKAQGCJFUHRZTIOLFAV`
Message #5: `OCIGPBPRPPAQUHPHRUEAVEJNPTHHKKBA`
Message #6: `YHMRLSXYYWCKWDXTIQMJONMPDEZFHPPB`
Message #7: `ZDJBDIRPTDLVRNNEQMKZSQRMQYFGEHNI`
Message #8: `KXAMXANRDSVYYXXCZKYSWAZRVFUHJLVM`
Message #9: `FPKSQEIURUYVHFUHTCMKCXJZYBHGELYD`
Message #10: `MEPXVPOLFZPEJWUWPIYSMFQHPNLKBYRJ`
Message #11: `XBWHENHQQLCPKOHJHSJPIXREOBWESMZK`
Message #12: `UFSWIYNEDCTVRFNWWAACKVOVGQLLSHNE`
Message #13: `CMOPRTIESCCRCACZGIMYNJQPVWLSWKLB`
Message #14: `LTCQZCWEGSXMKAGYZPJQFBZHQDHOKESR`
Message #15: `BATTHGBTDHIFQMGDAFUBOJBUUNMUWUWH`
Message #16: `SVGCAVVJFBJSKJHUYIMPUQFXKEMQFCDV`
Message #17: `DRBJFKFIRCVYCQTCXISQZGGVKRZPCOFC`
Message #18: `HIYVBUONZWWCUWVJIYTBJZLQYVVVCWKG`
Message #19: `PLQOMWBEGKREVJBKDGSVHPUNYTGNKQAH`
Message #20: `VZEISFETMYADXJNGFJLVCDYQGFQNWCVJ`
Message #21: `GQRDKJQUJKNKAJIRRPZCZGEKFVAFNGGN`
Message #22: `NNVLNLOJHPXXSKUAPSMVSSWEBUUUUJHK`
Message #23: `AWZEOZWRAPRZQUGXAQWXZNIZQALIVNJC`
Message #24: `JGNAGDPJFVASXLAISCPBPKZGWISZXINL`
Message #25: `WOXYTQIEIARLPNABWFVXNMRYTDRNMPAY`
Message #26: `EUFFWMDLTRMKZPPJZVPLGFVLXANJGDBO`

Determine what the day key is and then decrypt the remaining parts of ciphertext by using the message keys.

Eventually the Germans added more rotors and more wires for the plug board. Rejewski responded by building mechanical calculating machines, called "bombes", to break these harder versions of the Enigma machine. His methods continued to work because the Germans repeated the message key when sending a message. Eventually the Germans caught on and decided to no longer repeat message keys. Fortunately, Alan Turing of Bletchley Park (a cryptography think tank in Britain) was able to find another means of deducing the day key.

Turing's method of breaking the Enigma machine relied upon knowing parts of the plaintext. For instance, the German word for weather, which is `wetter`, would often appear at roughly the same place on messages containing a weather report. Sometimes, there would be a nice

cyclical relationship between the ciphertext characters for such a word. For instance, suppose that `wetter` encrypted to the word `TQRBUW`. This would mean that:

$$\texttt{w} \to \texttt{T}, \quad \texttt{t} \to \texttt{R}, \quad \texttt{r} \to \texttt{W}.$$

Composing these maps with each other gives a chain: $\texttt{W} \to \texttt{T} \to \texttt{R} \to \texttt{W}$.

If we were to encrypt the word `wetter` for various different rotor configurations, it would be relatively rare for such a randomly picked configuration to produce such a chain between the first, third, and sixth letters. Furthermore, since the particular configuration of the plug board had no affect upon whether such a chain existed or not, it would only require the checking of 105,391 possible rotor configurations. Like Rejewski, Turing built his own version of "bombes" to perform these computations.

We have only given a very brief summary of how Enigma was broken (following Singh's description of the events in Chapter 4 of *The Code Book*). It is important to note that our description of the breaking of Enigma has not been thorough and that there were many other techniques and people that contributed to the success of the allied forces decrypting Germany's messages.

4 Congruences and the Affine Cipher

We depart momentarily from our study of ciphers to look at some important concepts in number theory. Understanding such concepts will prove vital in understanding modern cryptosystems. We begin with a result which you probably first learned about in grade school.

Theorem 4.1 (Division Algorithm)**.** *Let m and n be integers. Suppose that n is non-zero. There exists unique integers q and r such that*

$$m = nq + r \tag{4.1}$$

and

$$0 \leq r < n. \tag{4.2}$$

Exercise 4.2. For $m = 17$ and $n = 5$, compute the q and r which occur in the Division Algorithm.

Exercise 4.3. Let $m = 21$ and $n = 7$. Since $21 = 7 \cdot 2 + 7$ does it follows that $q = 2$ and $r = 7$?

Exercise 4.4. Let $m = -13$ and $n = 6$. Compute q and r which occur in the Division Algorithm.

For m an integer and n a positive integer, define $m \,\%\, n$ to be the r which occurs in the Division Algorithm. We say $m \,\%\, n$ as "m mod n"'.

Exercise 4.5. Calculate the following:

(a) $73 \,\%\, 13$,

(b) $-57 \,\%\, 28$,

(c) $-3 \,\%\, 5$.

Let n and d be integers. If $n = cd$ for some integer c then we say that d *divides* n. Symbolically we write this as $d \mid n$. Such a d is called a *divisor* of n. If on the other hand we have that d is never a factor of n then we say that d *does not divide* n. Symbolically we write this as $d \nmid n$.

Exercise 4.6.

(a) Does $4 \mid 20$? (b) Does $3 \mid 5$? (c) Does $3 \mid 0$? (d) Does $0 \mid 3$? (e) Does $-2 \mid 4$?

If a and b are integers, n is a positive integer, and $n \mid a - b$, then we write that

$$a \equiv b (\mathrm{mod}\, n).$$

This sort of expression is called a *congruence*. It is not too hard to show that if $a \equiv b (\mathrm{mod}\, n)$ if and only if $a \,\%\, n = b \,\%\, n$.

Exercise 4.7. Determine whether the following statements are true or false.

(a) $4 \equiv 14 \pmod 5$ (b) $3 \equiv 12 \pmod 6$ (c) $-2 \equiv 12 \pmod 7$ (d) $16417 \equiv 2269 \pmod 7$

To get an intuitive feeling for when integers are congruent modulo n (i.e. when $a \equiv b (\mathrm{mod}\, n)$) consider the follow table:

⋮	⋮	⋮	⋮	⋮
-10	-9	-8	-7	-6
-5	-4	-3	-2	-1
0	1	2	3	4
5	6	7	8	9
10	11	12	13	14
15	16	17	18	19
⋮	⋮	⋮	⋮	⋮

Observe that numbers in the same column of the above table are congruent to each other modulo 5. Also notice that if two numbers are in different columns then those two numbers are not congruent to each other modulo 5. We let [0] denote the integers in the first column, [1] denote the integers in the second column, and so on until we end with writing [4] for the integers in the last column. We say that [0], [1], [2], [3], [4] are *congruence classes*. More generally, for a positive integer n and a integer j , we let $[j]$ denote the set of integers congruent to j modulo n. We denote the collection of all congruences classes for n by $\mathbb{Z}/n\mathbb{Z}$ (or if you're a topologist, by $\mathbb{Z}_n$). Thus

$$\mathbb{Z}/n\mathbb{Z} = \{[0], [1], \ldots, [n-1]\}.$$

As the following proposition shows, congruences behave very much like equalities; namely, many of the operations normally used in algebra to simplify equalities can also be used to simplify congruences.

Proposition 4.8. *Let a, b, c, d, n be integers with $n > 0$.*

(a) $a \equiv a (\mathrm{mod}\, n)$

(b) If $a \equiv b (\mathrm{mod}\, n)$ *then* $b \equiv a (\mathrm{mod}\, n)$.

(c) If $a \equiv b (\mathrm{mod}\, n)$ *and* $b \equiv c (\mathrm{mod}\, n)$ *then* $a \equiv c (\mathrm{mod}\, n)$.

(d) $a + c \equiv b + d (\mathrm{mod}\, n)$,

(e) $a - c \equiv b - d (\mathrm{mod}\, n)$,

(f) $a \cdot c \equiv b \cdot d (\mathrm{mod}\, n)$.

Proof. Since $a - a = 0 = 0 \cdot n$ then by definition part (a) follows.

If $a \equiv b (\mathrm{mod}\, n)$ then by definition it follows that there is an integer ℓ such that $a - b = \ell \cdot n$. When we multiply this equation by -1 we get that $b - a = (-\ell) \cdot n$ and so by definition $b \equiv a (\mathrm{mod}\, n)$. This proves part (b).

If $a \equiv b (\mathrm{mod}\, n)$ and $b \equiv c (\mathrm{mod}\, n)$ then it follows that there are integers ℓ_1 and ℓ_2 such that $a - b = \ell_1 n$ and $b - c = \ell_2 n$. Adding these two equations yields:

$$a - c = (a - b) + (b - c) = \ell_1 n + \ell_2 n = (\ell_1 + \ell_2) n.$$

So by definition $a \equiv c (\mathrm{mod}\, n)$. This proves part (c).

If $a \equiv b (\mathrm{mod}\, n)$ and $c \equiv d (\mathrm{mod}\, n)$ then it follows that there are integers ℓ_1, ℓ_2 such that $a - b = n\ell_1$ and $c - d = n\ell_2$. Adding these equations together shows that $a - b + c - d = n(\ell_1 + \ell_2)$. Rewriting the left hand side yields $(a + c) - (b + d) = n(\ell_1 + \ell_2)$. So by definition $a + c \equiv b + d (\mathrm{mod}\, n)$. This proves part (d). Part (e) is proved similarly.

Since $a - b = n\ell_1$ and $c - d = n\ell_2$ then $(a - b)c + b(c - d) = cn\ell_1 + bn\ell_2 = n(c\ell_1 + b\ell_2)$. Simplifying the left hand side yields $ac - bd = n(c\ell_1 + bp_1)$. So by definition $ac \equiv bd (\mathrm{mod}\, n)$. This proves part (f). □

With the above proposition we will show that it is possible to define addition for $\mathbb{Z}/n\mathbb{Z}$. Indeed, suppose we wished to add $[i], [j] \in \mathbb{Z}/n\mathbb{Z}$. How would we define it? One idea would to take an integer $x_1 \in [i]$ and an integer $y_1 \in [j]$ and define $[i] + [j] = [x_1 + y_1]$. But how do we know if we select a different $x_2 \in [i]$ and a different $y_2 \in [j]$ that we would get the same answer as we did with x_1 and y_1; namely, how do we know that $[x_1 + y_1] = [x_2 + y_2]$? The answer lies in part (d) of the above proposition. Indeed, if $x_1, x_2 \in [i]$ then by definition $x_1 \equiv x_2 (\mathrm{mod}\, n)$. Likewise, if $y_1, y_2 \in [j]$ then by definition $y_1 \equiv y_2 (\mathrm{mod}\, n)$. By part (d) it follows that $x_1 + y_1 \equiv x_2 + y_2 (\mathrm{mod}\, n)$ and so by definition $[x_1 + y_1] = [x_2 + y_2]$. Hence addition is indeed well-defined on $\mathbb{Z}/n\mathbb{Z}$. By a similar argument one can show that multiplication is also well-defined on $\mathbb{Z}/n\mathbb{Z}$. Since repeatedly writing square brackets can be cumbersome it is not uncommon for authors to simply write j for $[j]$.

Below is a correspondence between the 26 letters of the alphabet and the numbers 0-25:

a	b	c	d	e	f	g	h	i	j	k	l	m
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕
00	01	02	03	04	05	06	07	08	09	10	11	12

n	o	p	q	r	s	t	u	v	w	x	y	z
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕
13	14	15	16	17	18	19	20	21	22	23	24	25

(4.3)

A plaintext message that has been converted into numbers by (4.3) is called a *numeric plaintext*. A ciphertext message that has been converted into numbers by (4.3) is called a *numeric ciphertext*.

Now that we have our letters written in terms of numbers, we see that the shift cipher is simply involves taking a numeric plaintext number p and calculating the numeric ciphertext number $(p+k)\,\%\,26$ where k is our shift; or if we prefered to work in terms of $\mathbb{Z}/26\mathbb{Z}$, encryption of p involves simply calculating $[p]+[k]$.

Exercise 4.9. Consider the following plaintext:

`zebras zigzagging`

Convert this message to a numeric plaintext using (4.3) and then perform the Caesar cipher (i.e. the shift cipher for $k=3$) by sending each number p to $(p+3)\,\%\,26$. Then use (4.3) to compute the ciphertext in terms of letters.

By (4.3), we find that our numeric plaintext is:

25 04 01 17 00 08 25 08 06 25 00 06 06 08 13 06

Next we add 3 to each of these numbers:

28 07 04 20 03 11 28 11 09 28 03 09 09 11 16 09

For each of these numbers x we calculate $x\,\%\,26$. Except for $x=28$, all other x satisfy the inequality $0 \leq x < 26$. For these x we have that $x\,\%\,26 = x$. For $x=28$ we have that $28\,\%\,26 = (28-26)\,\%\,26 = 2\,\%\,26 = 2$. Thus we obtain:

02 07 04 20 03 11 02 11 09 02 03 09 09 11 16 09

By (4.3) we obtain the following ciphertext:

`CHEUDV CLJCDJJLQJ`

Alice wishes to send Bob a message. Prior to sending the message, Alice and Bob decide to use the affine cipher. Alice and Bob begin by agreeing upon a key. For the affine cipher, the key will consist of two integers a and b between 0 and 26 with the requirement that $a \neq 0, 2, 13, 26$.[3]

[3]We will explain this requirement later.

To encrypt a message, Alice converts her plaintext message to numeric plaintext and then for each number p of her numeric plaintext she computes $(a \cdot p + b) \,\%\, 26$. The resulting numbers will form her numeric ciphertext. She can then convert to an alphabetic ciphertext and send the message to Bob.

Exercise 4.10. Alice wishes to send Bob the following message:

`someone is spying on us`

Prior to sending the message, Alice and Bob decide to use the affine cipher with a key of $a = 5$ and $b = 2$ (notice that $\gcd(a, 26) = 1$). Encrypt this message.

We convert the plaintext message to the following numeric plaintext:

18 14 12 04 14 13 04 08 18 18 15 24 08 13 06 14 13 20 18

For each number p of the numeric plaintext, we compute $(5 \cdot p + 2) \,\%\, 26$. The following calculations show the arithmetic required to compute the first couple numbers of our numeric ciphertext:

$$(5 \cdot 18 + 2) \,\%\, 26 = 92 \,\%\, 26 = 14$$
$$(5 \cdot 14 + 2) \,\%\, 26 = 72 \,\%\, 26 = 20$$
$$(5 \cdot 12 + 2) \,\%\, 26 = 62 \,\%\, 26 = 10$$
$$(5 \cdot 4 + 2) \,\%\, 26 = 22 \,\%\, 26 = 22$$

Continuing, we will obtain the following numeric ciphertext:

14 20 10 22 20 15 22 16 14 14 25 18 16 15 06 20 15 24 14

The alphabetic ciphertext will then become:

`OUKWUPW QO OZSQPG UP YO`

Bob is then left with the task of decrypting the message. To do this, he will reference the following multiplication chart for $\mathbb{Z}/26\mathbb{Z}$:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
2	0	2	4	6	8	10	12	14	16	18	20	22	24	0	2	4	6	8	10	12	14	16	18	20	22	24
3	0	3	6	9	12	15	18	21	24	1	4	7	10	13	16	19	22	25	2	5	8	11	14	17	20	23
4	0	4	8	12	16	20	24	2	6	10	14	18	22	0	4	8	12	16	20	24	2	6	10	14	18	22
5	0	5	10	15	20	25	4	9	14	19	24	3	8	13	18	23	2	7	12	17	22	1	6	11	16	21
6	0	6	12	18	24	4	10	16	22	2	8	14	20	0	6	12	18	24	4	10	16	22	2	8	14	20
7	0	7	14	21	2	9	16	23	4	11	18	25	6	13	20	1	8	15	22	3	10	17	24	5	12	19
8	0	8	16	24	6	14	22	4	12	20	2	10	18	0	8	16	24	6	14	22	4	12	20	2	10	18
9	0	9	18	1	10	19	2	11	20	3	12	21	4	13	22	5	14	23	6	15	24	7	16	25	8	17
10	0	10	20	4	14	24	8	18	2	12	22	6	16	0	10	20	4	14	24	8	18	2	12	22	6	16
11	0	11	22	7	18	3	14	25	10	21	6	17	2	13	24	9	20	5	16	1	12	23	8	19	4	15
12	0	12	24	10	22	8	20	6	18	4	16	2	14	0	12	24	10	22	8	20	6	18	4	16	2	14
13	0	13	0	13	0	13	0	13	0	13	0	13	0	13	0	13	0	13	0	13	0	13	0	13	0	13
14	0	14	2	16	4	18	6	20	8	22	10	24	12	0	14	2	16	4	18	6	20	8	22	10	24	12
15	0	15	4	19	8	23	12	1	16	5	20	9	24	13	2	17	6	21	10	25	14	3	18	7	22	11
16	0	16	6	22	12	2	18	8	24	14	4	20	10	0	16	6	22	12	2	18	8	24	14	4	20	10
17	0	17	8	25	16	7	24	15	6	23	14	5	22	13	4	21	12	3	20	11	2	19	10	1	18	9
18	0	18	10	2	20	12	4	22	14	6	24	16	8	0	18	10	2	20	12	4	22	14	6	24	16	8
19	0	19	12	5	24	17	10	3	22	15	8	1	20	13	6	25	18	11	4	23	16	9	2	21	14	7
20	0	20	14	8	2	22	16	10	4	24	18	12	6	0	20	14	8	2	22	16	10	4	24	18	12	6
21	0	21	16	11	6	1	22	17	12	7	2	23	18	13	8	3	24	19	14	9	4	25	20	15	10	5
22	0	22	18	14	10	6	2	24	20	16	12	8	4	0	22	18	14	10	6	2	24	20	16	12	8	4
23	0	23	20	17	14	11	8	5	2	25	22	19	16	13	10	7	4	1	24	21	18	15	12	9	6	3
24	0	24	22	20	18	16	14	12	10	8	6	4	2	0	24	22	20	18	16	14	12	10	8	6	4	2
25	0	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

In referencing the above table, Bob sees that

$$5 \cdot 21 \equiv 1 (\mathrm{mod}\, 26).$$

If c is a numeric ciphertext number and p is the corresponding numeric plaintext number, then Bob knows that

$$c \equiv 5 \cdot p + 2 (\mathrm{mod}\, 26).$$

By multiplying both sides by 21, Bob finds that

$$21c \equiv 21 \cdot 5 \cdot p + 2 \equiv p + 2 (\mathrm{mod}\, 26)$$

and thus

$$p \equiv 21c - 2 (\mathrm{mod}\, 26).$$

Therefore, Bob can convert the numeric ciphertext he received from Alice to numeric plaintext by taking each numeric ciphertext number c and computing

$$p = (21c - 2) \,\%\, 26.$$

More generally, if a number c of the numeric ciphertext was computed from a number p of the numeric plaintext according to the formula:

$$c = (ap + b) \,\%\, 26,$$

we decrypt c by first finding d such that $a \cdot d \equiv 1 (\mathrm{mod}\, 26)$. For the time being this can be done easily enough by looking at the above table.[4] It then follows that by the same logic as earlier that

$$p = dc - p \,\%\, 26.$$

By using this formula, one can decrypt the affine cipher in general.

Exercise 4.11. Bob receives the following ciphertext from Alice:

```
KPVSPO C MKPBKPGK USBD C TXKTEMSBSEP SM MEAKBDSPO QT USBD UDSGD S USLL PEB TQB
```

When encrypting this message, Alice used a key of $a = 15$ and $b = 2$. Decrypt this message.

[4] Notice that there is no such b for $a = 0, 2, 13, 26$ - hence their exclusion when picking a initially.

5 The Extended Euclidean Algorithm

A positive integer p is a *prime number* if its only positive divisors are 1 and p. The first few primes are:

$$2, 3, 5, 7, 11, 13, 17, 19, 23, \ldots$$

Euclid showed that there are an infinite number of primes.

Theorem 5.1 (Unique Factorization)**.** *Each positive integer n can be written in a unique way as the product of prime numbers.*

We can also factor negative integers in a unique way as the product of prime numbers by factoring out a -1 and then factoring the resulting positive number.

Exercise 5.2. Factor the following integers as products of primes.

(a) 78 (b) 48 (c) -36 (d) 1 (e) 0

The *greatest common divisor* of two integers a and b is the largest positive integer dividing both a and b. Let $\gcd(a, b)$ denote this number. One way to compute $\gcd(a, b)$ is to first find the prime factorizations of both a and b.

Exercise 5.3. Calculate:

(a) $\gcd(48, -36)$ (b) $\gcd(30, 42)$ (c) $\gcd(17, 15)$

We say that two integers a and b are **relatively prime** if $\gcd(a, b) = 1$. Computing $\gcd(a, b)$ via the prime factorizations of a and b can sometimes be very difficult. This is due to the fact that in general, factoring integers can be very difficult.

Exercise 5.4. Compute the prime factorization of 48,970,477

Computing this prime factorization by hand would take a while because 48,970,477 has only large prime factors. In particular,

$$48,970,477 = 641 \cdot 317 \cdot 241.$$

Fortunately, there is a way to compute $\gcd(a, b)$ without knowing the prime factorizations of a and b.

Theorem 5.5 (Euclidean Algorithm)**.** *Let a and b be integers. Suppose $a > b$. Compute the integers q_1 and r_1 which satisfy:*

$$a = q_1 b + r_1 \quad \textit{and} \quad 0 \leq r_1 < b$$

(such integers are guaranteed to exist by the Division Algorithm). If $r_1 = 0$ then $a = q_1 b$ and $\gcd(a, b) = |b|$. Otherwise, compute the integers q_i and r_i which satisfy

$$\begin{aligned}
b &= q_2 r_1 + r_2 && 0 \leq r_2 < r_1 \\
r_1 &= q_3 r_2 + r_3 && 0 \leq r_3 < r_2 \\
r_2 &= q_4 r_3 + r_4 && 0 \leq r_4 < r_3 \\
&\vdots && \vdots \\
r_{k-1} &= q_{k+1} r_k + r_{k+1} && 0 \leq r_{k+1} < r_k
\end{aligned}$$

until $r_{k+1} = 0$ for some k; ***then***

$$\gcd(a, b) = r_k.$$

Exercise 5.6. Compute $\gcd(12345, 11111)$.

Let's work out the solution to this exercise. Let $a = 12345$ and $b = 11111$. Following the steps described in the Euclidean Algorithm, we compute the following:

$$\begin{aligned}
12345 &= (1) \cdot 11111 + 1234 \\
11111 &= (9) \cdot 1234 + 5 \\
1234 &= (246) \cdot 5 + 4 \\
5 &= (1) \cdot 4 + 1 \\
4 &= (4) \cdot 1 + 0
\end{aligned}$$

From this we see that $r_5 = 0$ and $r_4 = 1$. Therefore $\gcd(12345, 11111) = 1$.

Exercise 5.7. Compute $\gcd(32192, 8551)$.

Let's work out the solution to this exercise. Let $a = 32192$ and $b = 8551$. Following the steps described in the Euclidean Algorithm, we compute the following:

$$\begin{aligned}
32192 &= (3)8551 + 6539 \\
8551 &= (1)6539 + 2012 \\
6539 &= (3)2012 + 503 \\
2012 &= (4)503 + 0
\end{aligned}$$

From this we see that $r_4 = 0$ and $r_3 = 503$. Therefore $\gcd(32192, 8551) = 503$.

Exercise 5.8. Compute $\gcd(27554, 7321)$.

Let's work out the solution to this exercise. Let $a = 27554$ and $b = 7321$. Following the steps described in the Euclidean Algorithm, we compute the following:

$$\begin{aligned} 27554 &= (3)7321 + 5591 \\ 7321 &= (1)5591 + 1730 \\ 5591 &= (3)1730 + 401 \\ 1730 &= (4)401 + 126 \\ 401 &= (3)126 + 23 \\ 126 &= (5)23 + 11 \\ 23 &= (2)11 + 1 \\ 11 &= (1)11 + 0 \end{aligned}$$

From this we see that $r_8 = 0$ and $r_7 = 1$. Therefore $\gcd(27554, 7321) = 1$.

We have not yet addressed why the Euclidean algorithm does indeed give the $\gcd(a, b)$ or how long it takes to run. To see that the Euclidean algorithm does indeed give $\gcd(a, b)$ first notice that $\gcd(a, b) = \gcd(b, r_1)$. Indeed, any divisor of a and b must also divide r_1 since $r_1 = a - q_1 b$. Likewise, any divisor of r_1 and b must divide a since $a = q_1 b + r_1$. Thus $\gcd(a, b) = \gcd(b, r_1)$. The same argument can be applied to deduce that

$$\gcd(a, b) = \gcd(b, r_1) = \gcd(r_1, r_2) = \ldots = \gcd(r_{k-1}, r_k).$$

Since $\gcd(r_{k-1}, r_k) = r_k$ (recall that $r_{k+1} = 0$), it follows that the Euclidean algorithm does in fact give the $\gcd(a, b)$.

As for the running time of the Euclidean algorithm, this can be deduced from the following lemma which can be found in our textbook by Hoffstein, Pipher, and Silverman:

Lemma 5.9.

$$r_{i+2} < \frac{1}{2} r_i$$

Proof. If $r_{i+1} \leq \frac{1}{2} r_i$ then since $r_{i+2} < r_{i+1}$ it follows that $r_{i+2} < r_{i+1} \leq \frac{1}{2} r_i$ and then we are done. So instead, suppose $r_{i+1} > \frac{1}{2} r_i$. This inequality implies that $2 > \frac{r_i}{r_{i+1}}$. Since $r_{i+1} < r_i$ it follows that $\frac{r_i}{r_{i+1}} > 1$. Thus $2 > \frac{r_i}{r_{i+1}} > 1$. Therefore $r_i = r_{i+1} \cdot 1 + r_{i+2}$. Consequently,

$$r_{i+2} = r_i - r_{i+1} < r_i - \frac{1}{2} r_i = \frac{1}{2} r_i.$$

□

By repeatedly applying this lemma, we find that

$$0 = r_{k+1} < \frac{1}{2} r_{k-1} < \frac{1}{4} r_{k-3} < \frac{1}{8} r_{k-5} < \ldots < \frac{1}{2^{\lfloor k/2 \rfloor}} r_1 < \frac{1}{2^{\lfloor k/2 \rfloor}} b.$$

Thus

$$1 + \lfloor k/2 \rfloor < \log_2(b).$$

Hence the Euclidean algorithm takes at most about $\log_2(b)$ iterations to complete, which is really quite fast.

Theorem 5.10. *Let a and b be integers, not both equal to zero. There exists integers x and y such that*

$$ax + by = \gcd(a, b).$$

The following algorithm allows us to compute x and y for a given a and b.

Theorem 5.11 (Extended Euclidean Algorithm)**.** *Just as in the Euclidean Algorithm, compute the integers q_i and r_i which satisfy:*

$$\begin{aligned}
a &= q_1 b + r_1 && 0 \le r_1 < b \\
b &= q_2 r_1 + r_2 && 0 \le r_2 < r_1 \\
r_1 &= q_3 r_2 + r_3 && 0 \le r_3 < r_2 \\
r_2 &= q_4 r_3 + r_4 && 0 \le r_4 < r_3 \\
&\vdots && \vdots \\
r_{k-1} &= q_{k+1} r_k + r_{k+1} && 0 \le r_{k+1} < r_k
\end{aligned}$$

until $r_{k+1} = 0$ for some k. Let

$$\begin{aligned}
x_0 = 1, \quad x_1 = 0, \quad x_j = -q_{j-1} x_{j-1} + x_{j-2}, \\
y_0 = 0, \quad y_1 = 1, \quad y_j = -q_{j-1} y_{j-1} + y_{j-2},
\end{aligned}$$

then

$$ax_{k+1} + by_{k+1} = \gcd(a, b).$$

As an example, let's compute x and y such that $12345x + 11111y = \gcd(12345, 11111)$. Let $a = 12345$ and $b = 11111$. Previously when using the Euclidean Algorithm we computed the following:

$$\begin{aligned}
12345 &= (1) \cdot 11111 + 1234 \\
11111 &= (9) \cdot 1234 + 5 \\
1234 &= (246) \cdot 5 + 4 \\
5 &= (1) \cdot 4 + 1 \\
4 &= (4) \cdot 1 + 0
\end{aligned}$$

From this we see that $q_1 = 1$, $q_2 = 9$, $q_3 = 246$, $q_4 = 1$, and $q_5 = 4$. Thus

$$x_2 = -q_1 x_1 + x_0 = -1 \cdot 0 + 1 = 1 \qquad y_2 = -q_1 y_1 + y_0 = -1 \cdot 1 + 0 = -1$$

$$x_3 = -q_2x_2 + x_1 = -9 \cdot 1 + 0 = -9 \qquad y_3 = -q_2y_2 + y_1 = -9 \cdot (-1) + 1 = 10$$
$$x_4 = -q_3x_3 + x_2 = -246 \cdot (-9) + 1 = 2215 \qquad y_4 = -q_3y_3 + y_2 = -246 \cdot 10 - 1 = -2461$$
$$x_5 = -q_4x_4 + x_3 = -1 \cdot 2215 - 9 = -2224 \qquad y_5 = -q_4y_4 + y_3 = -1 \cdot (-2461) + 10 = 2471$$

Thus $x = x_5 = -2224$ and $y = y_5 = 2471$.

Exercise 5.12. Compute x and y such that $26x + 11y = \gcd(26, 11)$.

Exercise 5.13. Compute x and y such that $32192x + 8551y = \gcd(32192, 8551)$.

If there exists an integer y such that

$$ay \equiv 1 (\text{mod } n),$$

then we say that y is a *multiplicative inverse of a mod n*. There is no guarantee that such a congruence will always have a solution (try $a = 0$).

Exercise 5.14. Find an integer y such that $17y \equiv 1(\text{mod } 83)$; that is to say, find the multiplicative inverse of 17 mod 83. *(Hint: Try using the Extended Euclidean Algorithm.)*

Exercise 5.15. Find an integer y such that $153y \equiv 1(\text{mod } 2783)$; that is to say, find the multiplicative inverse of 153 mod 2783. *(Hint: Try using the Extended Euclidean Algorithm.)*

Exercise 5.16. Is there an easy way to tell if a does not have a multiplicative inverse mod n?

6 Exponentiation and Primitive Roots

Let n be a positive integer. We ask ourselves the following question: how many integers a are there such that $0 \leq a < n$ and $\gcd(n, a) = 1$? We let $\phi(n)$ denote the number of such a's. In the table below, we calculate these a's and $\phi(n)$:

n	a's	$\phi(n)$
2	1	1
3	1, 2	2
4	1, 3	2
5	1, 2, 3, 4	4
6	1, 5	2
7	1, 2, 3, 4, 5, 6	6
8	1, 3, 5, 7	4
9	1, 2, 4, 5, 7, 8	6
10	1, 3, 7, 9	4
11	1, 2, 3, 4, 5, 6, 7, 8, 9 10	10

The next theorem explains why we are interested in computing $\phi(n)$.

Theorem 6.1 (Euler's Theorem)**.** *If* $\gcd(n, a) = 1$ *then* $a^{\phi(n)} \equiv 1$ *(mod* n*).*

Let's check that this theorem is true for a few examples.

Exercise 6.2. Compute

(a) $3^4 \,\%\, 5$

(b) $7^6 \,\%\, 9$

(c) $9^{10} \,\%\, 11$

Euler's Theorem enable us to compute what would otherwise be very difficult exponentiation problems. For instance, suppose we wished to calculate the following:

$$6^{372893950372389271082} \,\%\, 11.$$

If we tried computing $6^{372893950372389271082}$ on a calculator or a computer, we would receive an error. This is due to the fact that $6^{372893950372389271082}$ is such a huge number that computers can not handle the computation. To calculate $6^{372893950372389271082} \,\%\, 11$ we instead first compute

$$372893950372389271082 \,\%\, \phi(11) = 372893950372389271082 \,\%\, 10.$$

A computer or calculator can easily compute that

$$372893950372389271082 \,\%\, 10 = 2.$$

This means that

$$372893950372389271082 = 10q + 2 = \phi(11) \cdot q + 2$$

for some integer q. Now we use Euler's theorem to observe that

$$6^{372893950372389271082} \equiv 6^{\phi(11)\cdot q+2} \equiv (6^{\phi(11)})^q \cdot 6^2 \equiv 1^q \cdot 6^2 \equiv 36 \pmod{11}.$$

Thus

$$6^{372893950372389271082} \,\%\, 11 = 36 \,\%\, 11 = 3.$$

This example illustrates the following consequence of Euler's Theorem:

Proposition 6.3. *If $e \equiv f \pmod{\phi(n)}$ then for a such that $\gcd(n, a) = 1$ we have that*

$$a^e \equiv a^f \pmod{n}.$$

Exercise 6.4. Compute the following:

(a) $5^{93423811} \,\%\, 8$

(b) $3^{217382912} \,\%\, 10$

Many nice formulas exist for computing $\phi(n)$. The next proposition gives two of these formulas:

Proposition 6.5.

(a) If p is a prime number then $\phi(p) = p - 1$.

(b) If $n = p \cdot q$ where p and q are odd primes such that $p \neq q$ then $\phi(n) = (p - 1)(q - 1)$.

Exercise 6.6.

(a) Compute $\phi(137)$.

(b) Compute $\phi(828263)$.

In some of the cryptographic algorithms we will study, we will need to exponentiate numbers with very large modulus (i.e. n is very large). Even though Euler's Theorem allows us to make the task of exponentiation easier, there are further simplifications we can do to increase the speed of our computations. The following example shows why even with Euler's Theorem, exponentiation can still be long and complicated. We will then show how to simplify this calculation.

Suppose we wished to compute the following:

$$73849^{1733} \,\%\, 10892767057.$$

Already our exponent of 1733 is a "small" number relative to 10892767057, so reducing mod $\phi(10892767057)$ is unlikely to simplify the exponent. If you attempt calculating 73849^{1733} on a calculator you will likely get an error due to the fact that this number has 8436 place values. To compute $73849^{1733} \,\%\, 10892767057$ quickly, we instead compute the following:

$$\begin{aligned}
73849^2 &\equiv 5453674801 \pmod{10892767057} \\
73849^4 &\equiv 5453674801^2 \equiv 3228239332 \pmod{10892767057} \\
73849^8 &\equiv 3228239332^2 \equiv 5418894004 \pmod{10892767057} \\
73849^{16} &\equiv 5418894004^2 \equiv 7304999075 \pmod{10892767057} \\
73849^{32} &\equiv 7304999075^2 \equiv 2024969615 \pmod{10892767057} \\
73849^{64} &\equiv 2024969615^2 \equiv 8294973030 \pmod{10892767057} \\
73849^{128} &\equiv 8294973030^2 \equiv 5712094508 \pmod{10892767057} \\
73849^{256} &\equiv 5712094508^2 \equiv 2587881710 \pmod{10892767057} \\
73849^{512} &\equiv 2587881710^2 \equiv 7590059148 \pmod{10892767057} \\
73849^{1024} &\equiv 7590059148^2 \equiv 2655009279 \pmod{10892767057}.
\end{aligned}$$

Since $1733 = 1024 + 512 + 128 + 64 + 4 + 1$ it follows that

$$\begin{aligned}
73849^{1733} &\equiv 73849^{1024+512+128+64+4+1} \\
&\equiv 73849^{1024} \cdot 73849^{512} \cdot 73849^{128} \cdot 73849^{64} \cdot 73849^{4} \cdot 73849^{1} \\
&\equiv 2655009279 \cdot 7590059148 \cdot 5712094508 \cdot 8294973030 \cdot 3228239332 \cdot 73849 \\
&\equiv 22763126378485995692865074127839955024144653212321344 0 \\
&\equiv 10010354795 \pmod{10892767057}
\end{aligned}$$

Exercise 6.7. Compute $3849^{1902} \,\%\, 10057$.

Exercise 6.8. Compute $27811^{79970} \,\%\, 79873$.

Consider the prime $p = 13$. In the following table, for each integer a such that $0 < a < p$, we calculate $a^e \,\%\, p$ for integers e such that $0 < e < p$.

	a^1	a^2	a^3	a^4	a^5	a^6	a^7	a^8	a^9	a^{10}	a^{11}	a^{12}
$a = 1$	1	1	1	1	1	1	1	1	1	1	1	1
$a = 2$	2	4	8	3	6	12	11	9	5	10	7	1
$a = 3$	3	9	1	3	9	1	3	9	1	3	9	1
$a = 4$	4	3	12	9	10	1	4	3	12	9	10	1
$a = 5$	5	12	8	1	5	12	8	1	5	12	8	1
$a = 6$	6	10	8	9	2	12	7	3	5	4	11	1
$a = 7$	7	10	5	9	11	12	6	3	8	4	2	1
$a = 8$	8	12	5	1	8	12	5	1	8	12	5	1
$a = 9$	9	3	1	9	3	1	9	3	1	9	3	1
$a = 10$	10	9	12	3	4	1	10	9	12	3	4	1
$a = 11$	11	4	5	3	7	12	2	9	8	10	6	1
$a = 12$	12	1	12	1	12	1	12	1	12	1	12	1

For some integers a we have that $a^1, a^2, \ldots a^{12} \bmod p = 13$ provides us with every integer b such that $0 < b < p$. In particular, $a = 2, 6, 7, 11$ have this property. The other values for a do not have this property. We give a special name to integers a which have the aforementioned property:

For a prime p, we say that a is a *primitive root mod p* if the numbers

$$a^1 \,\%\, p,\ a^2 \,\%\, p,\ \ldots,\ a^{p-1} \,\%\, p$$

give all the integers b such that $0 < b < p$. In the prior example, $a = 2, 6, 7, 11$ are primitive roots mod p.

Exercise 6.9. Find all the primitive roots a of $p = 11$ such that $0 < a < p$.

In Theorem 1.31 of our textbook, it is stated that for any choice of p, there exists a primitive root. More precisely, there are $\phi(p-1)$ such primitive roots.

7 Public Key Exchange Algorithms

Some of the most popular cryptosystems in use today is *AES (Advanced Encryption Standard)*, the successor to the still popular *DES (Digital Encryption Standard)* system. Algorithms such as AES and DES are *block ciphers*. A block cipher is simply a cipher that encrypts large chunks of text (or data) all at once, rather than one letter at a time. The RSA and ElGamal Algorithms, which we will talk about later, are examples of block cipher.s What distinguishes AES and DES from RSAand ElGamal is that AES and DES are optimized to encrypt and decrypt quickly. While RSA and ElGamal are fast enough to encrypt small amounts of data, they takes a noticeable amount of time to encrypt large amounts of data.

AES and DES are examples of *symmetric cryptosystems*;that is to say, the key which encrypts an AES or DES message is the same key that decrypts the message. The security of symmetric ciphers depend upon Alice and Bob keeping the key a secret from Eve. This leaves symmetric ciphers with a problem: How do Alice and Bob agree upon a key to use? This is called the *key distribution problem*. If Alice and Bob can only communicate with each other through public channels (such as is the case with Internet communication), then it appears that they must use some sort of public-key crypto-system (such as RSA) to first communicate a choice of key for the symmetric cipher and then encrypt the rest of their communication using the symmetric cipher. Indeed, many modern crypto-systems do precisely this.

Although algorithms such as RSA and ElGamal can be used to solve the problem of key distribution, these were not the first such solution. The credit for this accomplishment belongs to Diffie and Hellman and their algorithm known as the *Diffie-Hellman key exchange*. In this section we will explain how this algorithm operates.

(1) Alice and Bob select a "large" prime p and a primitive root α mod p (we will not discuss in detail how to find such a primitive root mod p; for now, we will just suppose that Alice and Bob have found by some means unknown to us such a primitive root α). They can communicate the choice of p and α to each other through public channels.

(2) Alice selects a random integer x such that $1 < x < p-2$, computes $\alpha^x \,\%\, p$, and sends this number to Bob. Alice's choice of x is a secret that she does not reveal to anyone.

(3) Bob selects a random integer y such that $1 < y < p-2$, computes $\alpha^y \,\%\, p$, and sends this number to Alice. Bob's choice of y is a secret that he does not reveal to anyone.

(4) Alice then calculates the number $K = (\alpha^y)^x \,\%\, p = \alpha^{xy} \,\%\, p$ and Bob computes $(\alpha^x)^y \,\%\, p = \alpha^{xy} \,\%\, p$ which is also equal to K.

Thus Alice and Bob have agreed upon a common number K and this number K can then be used as the key for a symmetric cipher.

Let's see how this calculation works out for a particular example. Suppose Bob picks $p = 25703$ and $\alpha = 5$ (I used Mathematica to check that $\alpha = 5$ really was a primitive root mod

$p = 25703$). Bob sends these numbers to Alice. Suppose Alice picks $x = 3881$ and computes that $\alpha^x \,\%\, p = 5^{3881} \,\%\, 25703 = 14976$. She then sends this integer to Bob. Suppose Bob picks $y = 11923$ and computes that $\alpha^y \,\%\, p = 5^{11923} \,\%\, 25703 = 13027$. He then sends this integer to Alice.

Next, Alice take the $\alpha^y \,\%\, p = 13027$ she received from Bob and computes $K = 13027^x \,\%\, p = 13027^{3881} \,\%\, 25703 = 10333$. Similarly, Bob takes the $\alpha^x \,\%\, p = 14976$ he received from Alice and computes $K = 14976^y \,\%\, p = 14976^{11923} \,\%\, 25703 = 10333$. We see then that Alice and Bob have obtained the same number K as we claimed earlier. They can now use this value of K for the key to some symmetric cipher.

Exercise 7.1. Suppose Alice and Bob wish to use the Diffie-Hellman key exchange. Suppose that Alice and Bob agree upon a prime $p = 29507047$ and a primitive root $\alpha = 3$. Suppose Alice selects $x = 172837$ and Bob selects $y = 8367282$. Compute K.

It is important to notice that neither Alice or Bob are able to directly pick the choice of K. They only pick the x and y, and from this the value of K is determined by arithmetic. Since Bob cannot know which random value of x Alice will pick, and since Alice cannot know which random value of y Bob will pick, neither of them can "rig" their selections of x and y so as to obtain a desired value for K. This could be considered a weakness for certain symmetric ciphers where not all keys are of equal strength (some speculate that this is the case for AES).

At this point it isn't entirely obvious that the Diffie-Hellman key exchange is secure. Suppose that Eve has intercepted all the communication between Alice and Bob. This means that Eve will know the values of p, α, $\alpha^x \,\%\, p$ and $\alpha^y \,\%\, p$. She only needs to calculate $K = \alpha^{xy}$ to decrypt Alice and Bob's communication via their chosen symmetric cipher. One way Eve could do this would be to somehow figure out the values of x and y. Once she knows one of these values it is then a simple matter to compute K from the information she already knows. The difficulty for Eve, is that computing such x and y strictly from p, α, $\alpha^x \,\%\, p$ and $\alpha^y \,\%\, p$ is extremely difficult. To illustrate this point, let's consider the following exercise:

Exercise 7.2. Let $p = 328543$, $\alpha = 3$ and $\alpha^x \,\%\, p = 102453$. Compute x.

One way of answering this problem is to do a brute-force like attack. That is, compute $\alpha^e \,\%\, p$ for various choices of e and see if eventually we obtain the value of 102453. If we did this approach, we wouldn't find a solution until we reached $e = 271891$. Obviously, it would take a great number of calculations to find this e and so it stands to reason that this approach is not very practical, especially if p is "large".

To the best of our knowledge, brute-force type attacks like these are the only means we have to solving this problem. The problem of computing x such that $\alpha^x \,\%\, p = m$ for given α, p, and m, is called the *discrete logarithm problem*. It is generally believed that the Diffie-Hellman key exchange is secure because the discrete logarithm problem is difficult to solve. This should

remind of you the RSA algorithm, which is also considered secure since factoring integers is believed to be difficult. Some of the chief rivals to the RSA algorithm, namely the ElGamal algorithm and elliptic curve algorithms, rely upon the discrete logarithm problem being difficult to solve.

Another algorithm that is also used for exchanging keys, and sometimes short messages, is the *Three-Pass Protocol*. Unlike with the Diffie-Hellman key exchange, Alice or Bob is free to pick a particular key to use. Here is how the algorithm works if Alice is to be the one selecting the key (we shall always assume this is the case):

(1) Alice picks a "large" prime p and shares this choice with Bob.

(2) Alice selects a random integer a such that $\gcd(p-1, a) = 1$. She then computes a^{-1} which will be an integer such that $a \cdot a^{-1} \equiv 1 \pmod{p-1}$. Alice then computes $K_1 = K^a \,\%\, p$ and sends this to Bob. Her choice of a (and equivalently a^{-1}) is a secret value only she knows.

(3) Bob selects a random integer b such that $\gcd(p-1, b) = 1$. He then computes b^{-1} which will be an integer such that $b \cdot b^{-1} \equiv 1 \pmod{p-1}$. Bob then computes $K_2 = K_1^b \,\%\, p$ and sends this to Alice. His choice of b (and equivalently b^{-1}) is a secret value only he knows.

(4) Alice receives K_2 from Bob, computes $K_3 = K_2^{(a^{-1})} \,\%\, p$, and then send this to Bob.

(5) Bob computes $K_3^{(b^{-1})} \,\%\, p$. This will be equal to K and so Bob will have received the key Alice wished to send.

Here is a quick explanation of why Bob should get K from this computation. Observe

$$K_3^{(b^{-1})} \equiv K_2^{(a^{-1})(b^{-1})} \equiv K_1^{b(a^{-1})(b^{-1})} \equiv K^{ab(a^{-1})(b^{-1})} \pmod{p}.$$

Recall from Euler's Theorem that if $e \equiv f \pmod{p-1}$ then $c^e \equiv c^f \pmod{p}$. Since

$$aba^{-1}b^{-1} \equiv aa^{-1}bb^{-1} \equiv 1 \cdot 1 \equiv 1 \pmod{p-1}$$

it follows that $K^{ab(a^{-1})(b^{-1})} \equiv K \pmod{p}$.

To see how this algorithm works, let's suppose that Alice picks a key of $K = 2382018$ and $p = 5848519$. Suppose she shares her value of p with Bob. Alice then has to pick an integer a such that $\gcd(p-1, a) = 1$. To find such an a, Alice can simply pick integers a at random until she finds one which satisfies $\gcd(p-1, a) = 1$. Let's suppose she pick $a = 382931$. Alice can then compute, using the Extended Euclidean Algorithm, that $a^{-1} = 4084895$. Alice then computes

$$K_1 = K^a \,\%\, p = 2382018^{382931} \,\%\, 5848519 = 2142641$$

and sends this value to Bob.

Bob picks an integer b such that $\gcd(p-1, b) = 1$. For instance, suppose Bob pick $b = 191005$. Bob then uses the Extended Euclidean Algorithm to compute that $b^{-1} = 2352145$. Next Bob computes

$$K_2 = K_1^b \,\%\, p = 2142641^{191005} \,\%\, 5848519 = 5039806$$

and sends this value to Alice.

Alice then computes

$$K_3 = K_2^{(a^{-1})} \,\%\, p = 5039806^{4084895} \,\%\, 5848519 = 5371106$$

and sends this to Bob. Finally, Bob computes

$$K_3^{(b^{-1})} \,\%\, p = 5371106^{2352145} \,\%\, 5848519 = 2382018$$

which is the value of K Alice had picked.

Exercise 7.3. Alice wishes to use the Three-pass protocol to communicate with Bob her choice of key for the AES algorithm. Suppose Alice wishes to use the key $K = 18291021$. Alice selects a prime $p = 49835497$ and random integer $a = 2215361$. Bob selects a random integer $b = 99215$. Confirm for this example that the Three-pass protocol really does result in Bob receiving K.

Exercise 7.4. Eve recorded the communication between Alice and Bob in the previous exercise. Alice decides she also wants to communicate with Eve using the AES algorithm. Alice decides to use the three pass protocol with the same values of p and a she used in the previous exercise (although she is careful to not use the same value of K). How can Eve trick Alice into giving her the key she used when she communicated with Bob in the previous exercise?

Like with the Diffie-Hellman key exchange, this algorithm is believed to be secure since the discrete logarithm problem is difficult to solve. Indeed, the only information that Eve is able to intercept through such a communication is the values of p, K_1, K_2, and K_3. The only obvious way for Eve to decrypt this message is to determine the values of a and b, which is equivalent to solving the discrete logarithm problem.

An obvious disadvantage for this algorithm is the need for multiple communications between Alice and Bob. However, it does have an advantage over the Diffie-Hellman key exchange in that Alice or Bob can pick any key they want. If Alice and Bob are using a symmetric cipher which may have "weak" keys, this might be an important feature to have.

8 The RSA Algorithm

The encryption methods we cover next will heavily use mathematics. Thus, we shall need a way to convert our plaintext messages to numerical sequences. We do this by use of the following rule:

a	b	c	d	e	f	g	h	i	j	k	l	m
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕
00	01	02	03	04	05	06	07	08	09	10	11	12

n	o	p	q	r	s	t	u	v	w	x	y	z
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕
13	14	15	16	17	18	19	20	21	22	23	24	25

(8.1)

So for example, if we had the plaintext message

`numbers`

it would convert to

13201201041718.

We call this string of numbers the *numeric plaintext*. We can similarly define the *numeric ciphertext* and use our rule (in reverse) to convert a numeric ciphertext to normal ciphertext.

In all of our cyrptosystems thus far, Alice and Bob have had to make sure that the key used in their encryption method is kept secret from Eve. In all of these algorithms the key for encryption and decryption were the same (or easily computed from each other). Such cryptosystems are called *symmetric key crypto-systems*.

The *RSA algorithm* differs from these encryption methods in that the encryption key is distinct from the decryption key. In fact, not only are these keys different, it is widely believed to be computationally unfeasible to determine the decryption key from the encryption key (or vice versa). Such systems are called an *asymmetric key cryptosystems*.

With the RSA algorithm, Bob is able to make the encryption key public to Alice (and Eve for that matter), Alice is able to encrypt her message and send it to Bob (with Eve possibly intercepting the ciphertext), and Bob is able to decrypt the message (without Eve being able to decrypt the message). Crypto-systems where Bob makes public the encryption key are called *public key crypto-systems*.

The RSA Algorithm begins with Bob performing the following steps:

(1) Select two "large"' prime numbers p and q. Let $n = p \cdot q$.

(2) Select an integer e such that $\gcd(\phi(n), e) = 1$. [Selecting e a prime number will usually work. Also recall that we can compute $\phi(n)$ easily by the following formula from our last lecture: $\phi(p \cdot q) = (p-1) \cdot (q-1)$ for p and q distinct, odd prime numbers.]

(3) Compute an integer d such that $d \cdot e \equiv 1 \pmod{\phi(n)}$. [We use the Extended Euclidean Algorithm to do this.]

(4) Bob makes public the integers e and n. He keeps d, p, and q secret.

As an example, let's suppose Bob selects primes $p = 16301$ and $q = 36709$. Bob computes $n = p \cdot q = 16301 \cdot 36709 = 598393409$. Observe that $\phi(n) = (16301-1)\cdot(36709-1) = 598340400$. Bob then selects an integer $e = 17389$ such that $\gcd(598340400, e) = 1$. Bob is then left with computing d such that $d \cdot 17389 \equiv 1 \pmod{598340400}$. To find d, Bob begins by performing the Euclidean Algorithm for $\gcd(598340400, 17389)$. Observe

$$\begin{aligned}
598340400 &= 17389(34409) + 2299 && q_1 = 34409,\ r_1 = 2299\\
17389 &= 2299(7) + 1296 && q_2 = 7,\ r_2 = 1296\\
2299 &= 1296(1) + 1003 && q_3 = 1,\ r_3 = 1003\\
1296 &= 1003(1) + 293 && q_4 = 1,\ r_4 = 293\\
1003 &= 293(3) + 124 && q_5 = 3,\ r_5 = 124\\
293 &= 124(2) + 45 && q_6 = 2,\ r_6 = 45\\
124 &= 45(2) + 34 && q_7 = 2,\ r_7 = 34\\
45 &= 34(1) + 11 && q_8 = 1,\ r_8 = 11\\
34 &= 11(3) + 1 && q_9 = 3,\ r_9 = 1\\
11 &= 1(11) + 0 && q_{10} = 11,\ r_{10} = 0
\end{aligned}$$

The Extended Euclidean Algorithm states that $598340400x_{10} + 17389y_{10} = 1$. Since

$$y_0 = 0, \quad y_1 = 1, \quad y_j = -q_{j-1}y_{j-1} + y_{j-2},$$

it follows that

$$\begin{aligned}
y_2 &= -q_1y_1 + r_1 = -34409(1) + 0 = -34409\\
y_3 &= -q_2y_2 + r_2 = -7(-34409) + 1 = 240864\\
y_4 &= -q_3y_3 + r_3 = -1(240864) + -34409 = -275273\\
y_5 &= -q_4y_4 + r_4 = -1(-275273) + 240864 = 516137\\
y_6 &= -q_5y_5 + r_5 = -3(516137) + -275273 = -1823684\\
y_7 &= -q_6y_6 + r_6 = -2(-1823684) + 516137 = 4163505\\
y_8 &= -q_7y_7 + r_7 = -2(4163505) + -1823684 = -10150694\\
y_9 &= -q_8y_8 + r_8 = -1(-10150694) + 4163505 = 14314199\\
y_{10} &= -q_9y_9 + r_9 = -3(14314199) + -10150694 = -53093291.
\end{aligned}$$

We could select -53093291 to be our d, but we prefer to have d be a positive integer. So instead we let $d = (-53093291)\,\%\,598340400 = 545247109$.

Bob then "publishes" $n = 598393409$ and $e = 17389$. Alice then performs the following steps to encrypt a message:

(1) Alice writes her numeric plaintext as one large number m. If $m > n$ then Alice cuts m up into pieces each of which is a number less than n. If this is the case then simply let m denote one of these pieces.

(2) Alice computes $c = m^e \,\%\, n$ and sends c to Bob. [We discussed how to perform this calculation efficiently in the previous lecture.]

(3) Alice sends c to Bob.

As an example, let's suppose that Alice wishes to send the following message:

`math`

Alice converts this plaintext message to numeric plaintext:

12 00 19 07

Alice can then convert this message to the number $m = 12001907$. To encrypt the message, Alice needs to compute $c = m^e \,\%\, 598393409 = 12001907^{17389} \,\%\, 598393409$. Notice that 17389 cannot be further reduced modulo $\phi(598393409) = 598340400$, so Euler's Theorem doesn't help us simplify this problem (although we will use this theorem later when decrypting). In order to compute $c = 12001907^{17389} \,\%\, 598393409$ we use the method of successive squaring. Observe

$$\begin{aligned}
12001907^2 &\equiv 510222169(\ \mathrm{mod}\ 598393409)\\
12001907^4 &\equiv 510222169^2 \equiv 164449803\ (\mathrm{mod}\ 598393409)\\
12001907^8 &\equiv 164449803^2 \equiv 434193028\ (\mathrm{mod}\ 598393409)\\
12001907^{16} &\equiv 434193028^2 \equiv 564311472\ (\mathrm{mod}\ 598393409)\\
12001907^{32} &\equiv 564311472^2 \equiv 481464120\ (\mathrm{mod}\ 598393409)\\
12001907^{64} &\equiv 481464120^2 \equiv 398840622\ (\mathrm{mod}\ 598393409)\\
12001907^{128} &\equiv 398840622^2 \equiv 486254146\ (\mathrm{mod}\ 598393409)\\
12001907^{256} &\equiv 486254146^2 \equiv 153391120\ (\mathrm{mod}\ 598393409)\\
12001907^{512} &\equiv 153391120^2 \equiv 270646901\ (\mathrm{mod}\ 598393409)\\
12001907^{1024} &\equiv 270646901^2 \equiv 319302272\ (\mathrm{mod}\ 598393409)\\
12001907^{2048} &\equiv 319302272^2 \equiv 396923525\ (\mathrm{mod}\ 598393409)\\
12001907^{4096} &\equiv 396923525^2 \equiv 150499076\ (\mathrm{mod}\ 598393409)\\
12001907^{8192} &\equiv 150499076^2 \equiv 442805031\ (\mathrm{mod}\ 598393409)\\
12001907^{16384} &\equiv 442805031^2 \equiv 103889026\ (\mathrm{mod}\ 598393409).
\end{aligned}$$

Since $17389 = 16384 + 512 + 256 + 128 + 64 + 32 + 8 + 4 + 1$ it follows that

$$\begin{aligned}
12001907^{17389} \equiv\ & 12001907^{16384} \cdot 12001907^{512} \cdot 12001907^{256} \cdot 12001907^{128} \cdot 12001907^{64}\\
& \cdot 12001907^{32} \cdot 12001907^{8} \cdot 12001907^{4} \cdot 12001907^{1}
\end{aligned}$$

$$\equiv (103889026 \cdot 270646901) \cdot (153391120 \cdot 486254146) \cdot (398840622 \cdot 481464120)$$
$$\cdot (434193028 \cdot 164449803) \cdot 12001907$$
$$\equiv (452778234 \cdot 257517887) \cdot (208469460 \cdot 88350933) \cdot 12001907$$
$$\equiv (484560428 \cdot 553522759) \cdot 12001907 \equiv 569086358 \cdot 12001907$$
$$\equiv 530804624 \pmod{598393409}.$$

Thus

$$c = 530804624.$$

Once Bob receives c, he can decrypt the message by doing the following steps:

(1) Calculate $m = c^d \,\%\, n$.

(2) Convert m to alphabetic plaintext.

For our example, we would need about 20 lines of computation to compute $c^d \,\%\, 598393409 = 530804624^{545247109} \,\%\, 598393409$ by use of successive squaring. Using a computer to calculate this instead, we find that

$$m = 530804624^{545247109} \,\%\, 598393409 = 12001907.$$

So we see that Bob does recover the original numeric plaintext and thus the original plaintext message.

Exercise 8.1. Alice encrypted a message using RSA:

$$7969085654.$$

Bob's public key is $n = 10669763123$ and $e = 50587$. Bob computed n by multiplying two prime numbers $p = 102061$ and $q = 104543$ together. Compute the decryption key d and decrypt this message. You may use the modular arithmetic applets for this problem.

Exercise 8.2. Alice wishes to send the following message to Bob using RSA:

`primes`

Bob's public key is $n = 3625543991239$ and $e = 26297$. Encrypt this message. You may use the modular arithmetic applets for this problem.

At this point there are two basic questions we should address:

(1) Why does the decryption method give us back the original message?

(2) Why is it (probably) computationally unfeasible for Eve to decrypt this ciphertext?

To answer the first question, we recall that our values for e and d were selected so that $e \cdot d \equiv 1 \pmod{\phi(n)}$. Thus there exists an integer q such that $e \cdot d = 1 + \phi(n) \cdot q$. Since $c = m^e \,\%\, n$ it follows then from Euler's theorem that:

$$c^d \equiv (m^e)^d \equiv m^{e \cdot d} \equiv m^{1+\phi(n)q} \equiv m \cdot (m^{\phi(n)})^q \equiv m \cdot 1^q \equiv m \pmod{n}.$$

[Technically the decryption process will only work for m such that $\gcd(n, m) = 1$. The only way this could not happen is if $m = p$ or $m = q$. Since it is exceedingly unlikely that m should equal p or q, we shall simply assume that $\gcd(n, m) = 1$.]

The second question is more complex. In order to decrypt the ciphertext, Eve must somehow calculate d. Recall that Bob calculated d by using the Euclidean Algorithm to compute $\gcd(\phi(n), e)$. Since Bob has already made the values of e and n public, Eve needs only to determine $\phi(n)$ to calculate d. The problem for Eve is that calculating $\phi(n)$ is equivalent to knowing how to factor n. To the best of our knowledge, it appears that factoring a large integer n (especially n that is the product of two large primes) is extremely difficult. The security of the RSA algorithm lies in the assumption that factoring such integers is computationally unfeasible.

Here are some additional RSA problems that are based off exercises from *Introduction to Cryptography with Coding Theory* by Trappe and Washington:

Exercise 8.3. By chance, Alice and Bob decide to both use $n = 156210463$ as their modulus for the RSA algorithm. Alice uses $e_A = 131849$ for her public encryption key and Bob uses $e_B = 58579$ for his public encryption key. Charlie wants to send a secret message m to both Alice and Bob. When he encrypts his message using Alice's key he obtain the following ciphertext:

$$c_A = 239590.$$

When he encrypts his message using Bob's key he obtains the following ciphertext:

$$c_B = 154226165.$$

Determine Charlie's original message m.

Exercise 8.4. Let $n = 341$ and $e = 17$. Alices decides to encrypt a message one letter at a time using the RSA algorithm. The following are the ciphertext numbers she receives:

$$2 \quad 172 \quad 138 \quad 16 \quad 53 \quad 59 \quad 52 \quad 16$$

Decrypt this message without factoring n.

Exercise 8.5. Your opponent uses RSA with $n = pq$ and encryption exponent e and encrypts a message m. This yields the ciphertext $c = m^e \,\%\, n$. A spy tells you that, for this message, $m^{12345} \equiv 1 \pmod{n}$. Describe how to determine m. For simplicity, assume $\gcd(12345, e) = 1$.

9 Primality Testing

In our discussion of the RSA algorithm, we have yet to explain how one important step of the algorithm is performed. Namely, in the first step of this algorithm Bob selects two "large" primes p and q. This begs the following question: how is Bob able to tell if a number is prime? At fist glance, it would appear that testing an integer for primality would be just as hard as factoring, but the surprising fact is that primality testing isn't actually that hard.

In this section we will discuss primality tests which tell us whether a particular integer n is *probably* prime. There are tests available that can determine with complete certainty whether n is prime, but these tests tend to be a bit more complicated. In light of this, we instead opt to show only probabilistic primality tests.

Fermat Primality Test: Let $n > 1$ be an integer. Pick a random integer a such that $1 < a < n - 1$. If $a^{n-1} \not\equiv 1 \pmod{n}$, then n is composite (i.e. not prime). If $a^{n-1} \equiv 1 \pmod{n}$ then n is *probably* prime.

Exercise 9.1. Use the Fermat primality test to determine if the following integers are composite or probably prime:

(a) $n = 561$ using $a = 2$

(b) $n = 38$ using $a = 5$

(c) $n = 174440043$ and $a = 7$,

(d) $n = 52711$ and $a = 2$.

Notice that the Fermat primality test doesn't guarantee that a given integer is prime, it merely states that it is *probably* prime. It is possible for the Fermat primality test to say that a composite number is probably prime. For example, $n = 561$ is a composite number, yet for part (a) of the previous exercise, the Fermat primality test said it was probably prime. If a composite integer n "passes" the Fermat primality test for a particular a then we say that n is a *pseudoprime for the base* a. Experimental evidence shows that pseudoprimes are fairly rare. If $n \leq 10^{10}$ and passes the Fermat primality test then there is only a probability of $3/100,000$ that n will be a pseudoprime for $a = 2$. The probabilities for other choices of a are also low.

Fermat's primality test is a simple consequence of Euler's Theorem. Recall that if we apply Euler's Theorem to n a prime number then for $1 < a < n$ we have that

$$a^{n-1} \equiv 1 \pmod{n}$$

since $\phi(n) = n - 1$. Therefore the only way that we could have

$$a^{n-1} \not\equiv 1 \pmod{n}$$

is if n were a composite number. This is essentially the logic behind the Fermat primality test.

The following test is a generalization of Fermat's primality test:

Proposition 9.2 (Miller-Rabin Primality Test)**.** *Let $n > 1$ be an odd integer. Therefore $n-1 = 2^k m$ where $k \geq 1$ and m is odd. Pick an integer a such that $1 < a < n-1$. Let*

$$\begin{aligned} b_0 &\equiv a^m \pmod{n} \\ b_1 &\equiv b_0^2 \pmod{n} \\ b_2 &\equiv b_1^2 \pmod{n} \\ &\vdots \\ b_{k-1} &\equiv b_{k-2}^2 \pmod{n}. \end{aligned}$$

If $b_0 \equiv \pm 1 \pmod{n}$ then n is probably prime.
Otherwise consider b_1.
If $b_1 \equiv 1 \pmod{n}$ then n is composite.
If $b_1 \equiv -1 \pmod{n}$ then n is probably prime.
If b_1 is congruent to neither ± 1 then consider b_2.
If $b_2 \equiv 1 \pmod{n}$ then n is composite.
If $b_2 \equiv -1 \pmod{n}$ then n is probably prime.
If b_2 is congruent to neither ± 1 then consider b_3. Continue in this way until either we declare n composite or probably prime, or until we are forced to consider b_{k-1}. In the latter case, we conclude the test by saying:
If $b_{k-1} \equiv -1$ then n is probably prime.
If $b_{k-1} \not\equiv -1$ then n is composite.

Let's perform the Miller-Rabin test for $n = 1801$ using $a = 5$. We begin by factoring out powers of 2 in $n - 1 = 1800$. One can easily compute that $n - 1 = 1800 = 2^3 \cdot 225$. Thus $k = 3$ and $m = 225$. We used the method of successive squares to compute the following exponentials. Observe:

$$b_0 \equiv 5^{225} \equiv 977 \pmod{1801}.$$

Thus we cannot conclude (yet) that n is composite or *probably* prime. Next observe:

$$b_1 \equiv 977^2 \equiv -1 \pmod{1801}.$$

We conclude then that $n = 1801$ is *probably* prime.

Next, let's perform the Miller-Rabin test for $n = 561$ using $a = 2$. We begin by factoring out powers of 2 in $n - 1 = 560$. One can easily compute that $n - 1 = 2^4 \cdot 35$. Thus $k = 4$ and $m = 35$. Observe

$$b_0 \equiv 2^{35} \equiv 263 \pmod{561}.$$

We cannot conclude (yet) that n is composite or *probably* prime. Next observe:

$$b_1 \equiv 263^2 \equiv 166 \pmod{561}.$$

We cannot conclude (yet) that n is composite or *probably* prime. Next observe:

$$b_2 \equiv 166^2 \equiv 67 \pmod{561}.$$

We cannot conclude (yet) that n is composite or *probably* prime. Next observe:

$$b_3 \equiv 67^2 \equiv 1 \pmod{561}.$$

Since our $k = 4$ it follows then that $n = 561$ is composite.

Exercise 9.3. Use the Miller-Rabin primality test to determine if the following integers are composite or probably prime:

(a) $n = 561$ and $a = 2$,

(b) $n = 25221$ and $a = 4$,

(c) $n = 5691041$ and $a = 3$,

(d) $n = 4962215809$ and $a = 5$.

If a composite integer n "passes" the Miller-Rabin test for some a then we say that n is a *strong pseudoprime for base a*. If $n \leq 10^{10}$ and passes the Miller-Rabin primality test for $a = 2$ then there is only a probability of $7/1,000,000$ that n will be a pseudoprime for $a = 2$. The probabilities for other choices of a are also low. From part (a) of the previous exercises, we see that the Miller-Rabin test really is a stronger test than the Fermat test since with the Miller-Rabin test 561 is now correctly classified as composite.

In practice, an integer n which passes the Miller-Rabin test for a particular a will then also be tested once again for another choice of a. An integer which passes the Miller-Rabin test for 10 choices of a has at most a probably of $1/1,000,000$ of being a composite. The accuracy of the Miller-Rabin test is owed in part to a result of Miller which asserts that if the Generalized Riemann Hypothesis is true, then one can compute a relatively small value r such that if an integer n passes the Miller-Rabin test for all $a \leq r$, then n is definitely prime.[5]

Now we explain how Bob picks p and q. Bob begins by randomly selecting a large integer t. He then applies primality tests to determine if t is (probably) prime. If t passes these tests, then Bob lets p or q be this t. If t is not prime, then Bob picks another large integer at random.[6] Once again, Bob applies his primality tests and continues picking random large integers should these test fails.

This leads to the following question: How likely is Bob to pick a prime at random? If this probability is very low, then it may take Bob a very long time to find primes p and q, and this

[5]The Generalized Riemann Hypothesis (along with its "simpler" version known as as the Riemann Hypothesis) are extremely difficult problems to solve. The Clay Mathematics Institute is offering one million dollars to whoever proves it first (there is no monetary award for disproving it).

[6]Bob can also simply increment his previous choice of t by 2 to get another large integer that might be prime.

could make the RSA algorithm too difficult to use. Fortunately, Bob is likely to pick a prime number rather quickly. This is due to a result known as the Prime Number Theorem. According to the Prime Number Theorem, the probability that an integer n is prime is simply $1/\ln(n)$. If we make sure to not select even numbers, then this probability reduces to $1/(2\ln(n))$. If n were to have 200 digits, the probability that n is prime is roughly $1/230$. Thus we can expect to find a prime number after 115 guesses. If Bob were doing this computation by hand, this would take very long. But fortunately computers are capable of doing this computation extremely quickly, making the selection of primes rather easy in application.

10 Digital Signatures and Factorization Algorithms

Public key algorithms such RSA appear to provide Alice and Bob with a great amount of security since they are related to difficult to solve math problems. But there is a relatively simple way that Eve can hinder effective communication between Alice and Bob. Let's suppose that Bob has a public key available so that Alice or anyone of Bob's other friends, can send him encrypted messages. Since this key is public, Eve also has access to it, so she can send messages to Bob as well. In fact, Eve can send messages to Bob and make these messages appear as though they were sent from Alice (this isn't too hard to do in practice with email). Thus, Eve is able to disrupt communication between Alice and Bob by impostering Alice. What Bob needs is a way to confirm that a message sent by Alice was actually sent by Alice. To do this, we will need what are called *digital signatures*.

Below is a description of what we call the *RSA digital signature scheme*:

(1) Alice picks two large primes p and q and computes $n = p \cdot q$. Alice then picks an integer e such that $\gcd(\phi(n), e) = 1$ and calculates d such that $e \cdot d \equiv 1 \pmod{\phi(n)}$. Alice publishes e and n and keeps the other quantities she computed secret.

(2) If m is a message which Alice wishes to sign, then Alice computes

$$y = m^d \,\%\, n$$

and send (m, y) to Bob.

(3) Bob downloads Alice's public key e and n and computes:

$$z = y^e \,\%\, n.$$

If $z = m$ then Bob accepts Alice's signature as valid.

Why is it that Bob accepts the signature if $z = y^e \,\%\, n$? Recall that Euler's Theorem tells us that if $s \equiv t \pmod{\phi(n)}$ then

$$x^s \equiv x^t \pmod{n}.$$

Therefore, since $e \cdot d \equiv 1 \pmod{\phi(n)}$ then

$$y^e \equiv (m^d)^e \equiv m^{de} \equiv m \pmod{n}.$$

This signature scheme is secure for the same reason the RSA algorithm is secure: ultimately, Eve would need to compute d from n and e in order to forge Alice's signature, and this is believed to be just as difficult as factoring n (which is extremely difficult to do).

Let's look at an example. Suppose that Alice has selected $p = 26407$ and $q = 33617$ and computes $n = p \cdot q = 887724119$. Notice that $\phi(n) = 26406 \cdot 33616 = 887664096$. Next Alice selects $e = 76421$ and then computes the integer d such that

$$e \cdot d \equiv 1 \pmod{\phi(n)}.$$

Alice computes that $d = 111357293$. Suppose that Alice wishes to send the following message:

`yes`

Converting to numeric plaintext, she writes this message as $m = 240418$. Alice then computes that

$$y = m^d \,\%\, n = 240418^{111357293} \,\%\, 887724119 = 98253548.$$

and sends m and y to Bob.

Next, Bob downloads Alice's public key: e and n. He then computes that

$$z = y^e \,\%\, n = 98253548^{76421} \,\%\, 887724119 = 240418.$$

Since this agree with m, Bob concludes that Alice really did send this message.

Exercise 10.1. Alice selects prime numbers $p = 10289$ and $q = 7759$ and lets $n = p \cdot q$. She selects $e = 4787$.

(a) Calculate an integer d such that $e \cdot d \equiv 1 \pmod{\text{n}}$.

(b) Sign the message `buy` using d. Check that the public key e actually verifies this signature.

Suppose that Bob has found a proof of the Riemann Hypothesis. Bob writes down his solution but decides he doesn't want to share his proof with the world quite yet. He think he can also prove the Birch-Swinnerton-Dyer Conjecture using the techniques in his proof and he doesn't want to have someone else beat him to the proof by taking advantage of his techniques. Yet there is always the chance that someone else might prove the Riemann Hypothesis before he has publicly released his proof, thus making him ineligible for the glory of being the first to solve the problem. Bob desires to be able to claim he was the first to prove the Riemann Hypothesis without making the proof public. To do this, he will use the following variant of the RSA digital signature scheme which is known as the *RSA blind signature scheme*:

(1) Alice selects distinct prime numbers p and q and computes $n = p \cdot q$. She selects an integer e such that $\gcd(\phi(n), e) = 1$ and computes d such that $e \cdot d \equiv 1 \pmod{\phi(n)}$. She makes n and e public.

(2) Bob picks a random integer k such that $\gcd(n, k) = 1$ and computes $t = (k^e \cdot m) \,\%\, n$ where m is his message. Bob sends t to Alice.

(3) Alice computes $s = t^d \,\%\, n$ and sends s back to Bob.

(4) Bob computes k^{-1} such that $k \cdot k^{-1} \equiv 1 \pmod{n}$ and calculates $r = (s \cdot k^{-1}) \,\%\, n$. One can show that $r = m^d \,\%\, n$.

(5) If Bob wishes to verify to someone that he sent m to Alice and that Alice signed it, he need only compute $r^e \,\%\, n$ which will be equal to m.

Notice that in this algorithm, Alice is unable to deduce the original message m. This is because Bob has multiplied m by k^e which is a random integer. If Eve is listening in on the communication she only knows the values of n, e, t, s, and r. The values of t and s have been "polluted" by the k^e multiplication preformed earlier, so they aren't very useful to Eve for determining m. Thus to read m Eve would need to compute $r^d \,\%\, n$, which is equivalent to Eve breaking the usual RSA algorithm.

In step (4) we made the statement that

$$(s \cdot k^{-1}) \,\%\, n = r = m^d \,\%\, n.$$

To see that this is true observe that

$$s \cdot k^{-1} \equiv t^d \cdot k^{-1} \equiv (k^e m)^d k^{-1} \equiv m^d k^{ed} k^{-1} \pmod{\phi(n)}.$$

Since $e \cdot d \equiv 1 \pmod{\phi(n)}$ it follows from Euler's Theorem that:

$$k^{ed} \cdot k^{-1} \equiv k \cdot k^{-1} \equiv 1 \pmod{n}.$$

Thus

$$s \cdot k^{-1} \,\%\, n = m^d \,\%\, n$$

as claimed.

Using this particular signature scheme, Bob is able to have Alice confirm that Bob had a solution to the Riemann Hypothesis without having actually reading Bob's message. In general, these types of digital signature schemes are called *blind signature schemes*.

Let's consider the following example. Suppose Alice has selected $p = 14759$ and $q = 57383$. Thus $n = p \cdot q = 846915697$. Suppose Alice selects $e = 10009$. She then computes d such that $e \cdot d \equiv 1 \pmod{\phi(n)}$. Alice can easily do this since she knows that $\phi(n) = (p-1) \cdot (q-1) = 846843556$. Alice finds that $d = 671704565$.

Suppose Bob wishes to have Alice sign the following message without reading it:

`zeta`

Bob converts this message to the integer $m = 25041900$. He selects a random integer $k = 117271$, checks that $\gcd(n, k) = 1$, and computes

$$k^e \cdot m \equiv 117271^{10009} \cdot 25041900 \equiv 719651529 \cdot 25041900 = 423846224 \pmod{n}$$

Bob then sends $t = 423846224$ to Alice. Alice then computes

$$s = t^d \,\%\, n = 423846224^{671704565} \,\%\, 846915697 = 599655035$$

and sends this to Bob.

Upon receiving s, Bob computes an integer k^{-1} such that $k \cdot k^{-1} \equiv 1 \pmod{n}$. Bob computes that $k^{-1} = 107555282$. Next, Bob computes

$$r = (s \cdot k^{-1}) \,\%\, n = (599655035 \cdot 107555282) \,\%\, 846915697 = 128612141.$$

If Bob wishes to show someone that Alice really did sign his message m, Bob only needs to compute

$$r^e = 128612141^{10009} \,\%\, 846915697 = 25041900$$

and this is indeed equal to m.

Next we discuss some of the attacks available against the RSA algorithm and its variants. As we've already mentioned, the strength of the RSA algorithm stems from the difficulty in factoring integers. So it follows that factoring algorithms are the first tool consider when attacking RSA.

Recall that with the RSA algorithm, our integer n is the product of two large prime numbers p and q. If we let $x = \frac{p+q}{2}$ and $y = \frac{p-q}{2}$ then it follows that

$$n = (x - y) \cdot (x + y) = x^2 - y^2.$$

Consequently, the task of factoring n reduces down to finding integers x and y such that $n = x^2 - y^2$. Equivalently, factorization boils down to computing $n + y^2$ for various values of y until we find $n + y^2$ that is a square. This method of factorization is known as *Fermat factorization*.

Exercise 10.2. Use Fermat factorization to factor the following integers:

(a) 295,927

(b) 58,167,792,391

(c) 2183

As the following exercise illustrates, the speed of the Fermat factorization is directly determined by how close the prime factors are to each other. In other words, it is easy to factor n when the factors p and q are close to each other. Because of this, most implementations of the RSA algorithm make sure to pick p and q that are sufficiently far apart so as to make the Fermat factorization attack impractical.

Another popular factorization algorithm is the *Pollard $p-1$ algorithm*.

Proposition 10.3 (Pollard $p-1$ Algorithm)**.** *Let n be an integer we wish to factor. Choose an integer $a > 1$ and a large interger B. Let $b = a^{B!} \pmod{n}$ and $d = \gcd(b-1, n)$. If $1 < d < n$ then d is a factor of n.*

As an example, let's apply the Pollard $p-1$ algorithm to factoring $n = 659635$. Suppose we pick $a = 2$ and $B = 10$. We can compute b in the following manner: Let

$$b_1 = a\%n$$

$$
\begin{aligned}
b_2 &= b_1^2 \% n \\
b_3 &= b_2^3 \% n \\
b_4 &= b_3^4 \% n \\
&\vdots
\end{aligned}
$$

Notice that it follows that $b = b_B$. When we perform this computation for our particular example we find that $b = 398071$. Next, we use the Euclidean algorithm to compute $\gcd(b-1, n) = \gcd(398070, 659635) = 5$. It is then easy to see from the Pollard $p-1$ algorithm that 5 is a factor n.

Exercise 10.4. Use the Pollard $p-1$ algorithm to factor the following integers (you are free to make your own choices for a and B).

(a) 85,979

(b) 47,306,490

(c) 84,338,552,447

11 Other Ciphers

Next we describe the *One-Time Pad* cipher. This cipher has the distinction of being perfectly secure and somewhat impractical. Suppose Alice wants to send the following message to Bob:

```
buy my pottery
```

Notice that this message consists of 12 letters. To encrypt this message, Alice uses a 12 letter, randomly generated string, such as the following:

```
JWBGISKJCXHA
```

Recall the following letter to number correspondence we used earlier:

A	B	C	D	E	F	G	H	I	J	K	L	M
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕
00	01	02	03	04	05	06	07	08	09	10	11	12

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕
13	14	15	16	17	18	19	20	21	22	23	24	25

According to this correspondence, `J` (the first letter of her random string) corresponds to 9. She then shifts the first letter of plaintext, `b`, by 9 characters. This will result in her encrypting `b` to `K`. Next, Alice observes that the second letter of her random string, `W`, corresponds to 22. Consequently, Alice then shifts the second letter of her plaintext, `u`, by 22 characters. This will result in her encrypting `u` to `Q`. Continuing in this way, Alice obtains the following ciphertext:

```
KQZSGHXCVBYY
```

Bob can easily decrypt this ciphertext provided he also knows the key that was used; namely the random sequence `JWBGISKJCXHA`. Thus in order to use the one-time pad cipher, Alice and Bob must have identical randomly generated sequences.

Exercise 11.1. The ciphertext

```
CRVFKIQCASVUMEDEV
```

was encrypted using the one-time pad cipher with the following key:

```
VDHOKKLOJGVBFCDSG
```

Decrypt this message.

Exercise 11.2. Alice and Eve are at war. Eve has intercepted the following messages Alice sent to her ally Bob:

```
CNPESTOWVSHYOQRHKV
```

and

```
KQEPBWOAHTXWEKNMS
```

In addition to obtaining these ciphertext, Eve learns that the messages were encrypted using the one-time pad cipher and that Alice used the same key for both messages. Eve has a strong suspicion that the plaintext for the first ciphertext begins with the word `attack`. Using this information, decrypt these messages.

As the prior exercise shows, it is vital to use a different key for each encryption when using the one-time pad cipher. Thus, if Alice and Bob wish to communicate with each other often, they must share a large common supply of randomly generated strings. Prior to computer communication it was logistically difficult to ensure that such a supply of randomly generated strings could be provided, especially during war. Consequently, the one-time pad cipher was rarely used prior to the use of computers. With the advent of computers, there came the ability to generate so called *pseudo-random* sequences, and with them, the ability to easily implement ciphers that are very similar to the one-time pad.

Next, we describe the *Blum-Blum-Shub Algorithm*, an algorithm for generating pseudo-random strings of bits (i.e. 0's and 1's). Here are the steps to the algorithm:

(1) Select two large primes p and q such that $p \equiv q \equiv 3 \pmod 4$. Let $n = p \cdot q$.

(2) Select a random integer x that is relatively prime to n. We shall refer to x as the *seed*.

(3) Let $x_0 = x^2 \,\%\, n$, and for each positive integer j (up to some bound), let $x_j = x_{j-1}^2 \,\%\, n$.

(4) Let b_j be the least significant bit of x_j. The string

$$b_1 b_2 b_3 \ldots b_j \ldots$$

is our pseudo-random bit string.

The requirement that $p \equiv q \equiv 3 \pmod 4$ helps to ensure that the sequence really is very nearly random. As an example, suppose we pick $p = 9043$ and $q = 8243$ (one can check that this choice of p and q satisfy the congruence stated in (1)). Thus $n = p \cdot q = 74541449$. Let $x = 339283$ (one can check that this choice of x is indeed relatively prime to n). We perform the following computations:

$$x_0 \equiv x^2 \equiv 20956833 \pmod{74541449}$$

$$x_1 \equiv x_0^2 \equiv 24104463 \pmod{74541449}$$
$$x_2 \equiv x_1^2 \equiv 34738927 \pmod{74541449}$$
$$x_3 \equiv x_2^2 \equiv 11663236 \pmod{74541449}$$
$$x_4 \equiv x_3^2 \equiv 20956833 \pmod{74541449}$$
$$x_5 \equiv x_4^2 \equiv 11004351 \pmod{74541449}$$

By converting 3, 3, 7, 6, 3, and 1 to binary, we see that

$$b_0 = 1, \quad b_1 = 1, \quad b_2 = 1, \quad b_3 = 0, \quad b_4 = 1, \quad b_5 = 1.$$

Thus our pseudo-random bit sequence is $111011\ldots$.

When Alice writes a message on her computer, that message is stored as a sequence of 0's and 1's. In particular, suppose that Alice's message (when written in binary) consists of the following strings of bits:

$$00101111100010100111.$$

When Alice runs Blum-Blum-Shub for $n = 74541449$ and $x = 339283$, she will obtain the following string of bits:

$$11011011100110110010.$$

To encrypt Alice's message using this random bit string, we need to define the XOR operation (a.k.a *exclusive or*) of these two bit strings. We denote the XOR operator by $\oplus$ and define it as follows:

$$0 \oplus 0 = 0, \qquad 0 \oplus 1 = 1, \qquad 1 \oplus 0 = 1, \qquad 1 \oplus 1 = 0.$$

The $\oplus$ operation extends in the obvious ways to strings of bits. For example, observe:

$$011 \oplus 101 = 110$$
$$01001110 \oplus 11011100 = 10010010$$
$$11110101 \oplus 00100110 = 11010011.$$

When we apply $\oplus$ to 00101111100010100111 and 11011011100110110010, we obtain the following:

$$11110100000100010101.$$

This string of bits would be Alice's ciphertext using Blum-Blum-Shub.

Exercise 11.3. Bob receives the ciphertext 11110100000100010101. How does he decrypt this ciphertext (we will assume that Bob knows only n and x).