Cryptography

Tom Davis
tomrdavis@earthlink.net
http://www.geometer.org/mathcircles

February 7, 2000

1 Introduction

The goal of cryptography is to make it possible for two people to exchange a message in such a way that other people cannot understand the message. There is no end to the number of ways this can be done, but here we will be concerned with methods of altering the text in such a way that the recipient can undo the alteration and discover the original text.

The original text is usually called "cleartext" and the encoded or altered text is called "ciphertext". The conversion from cleartext to ciphertext is called "encoding" or "enciphering", and the opposite operation is called "decoding" or "deciphering". If you are trying to read a secret message that was not intended for you and you initially don't know the encoding method, it is called "cracking" the code.

In general, the more ciphertext you have, the easier it is to crack the code. So generally it is a good idea to change the coding mechanism regularly. For example, if a coding scheme has a keyword (like the Vigenère cipher described below), if a different keyword is used every day, there may never be enough ciphertext to decode the message. But if you change the encoding every day, you need to have some method of getting the new keyword to the intended recipient in a secure way. The easiest way to crack a code is to steal the codebook!

There is a way around this that we'll discuss in the section on public key cryptography.

2 Simple Substitution Ciphers

In a simple substitution cipher, one character is substituted for another. Here is a simple example:

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

To encode some text, simply find each character in the text in the first line, and replace it by the character below it. For example, using the example above, if you encode the word "BIRDBRAIN", you get "ZPOUZORPS". To decode, reverse the process—for the first character in "ZPOUZORPS", find "Z" in the lower line, look above it to get "B"—the first letter of "BIRDBRAIN", et cetera.

If you have to decode a lot, it is easier if you invert the line above to get the table below. With this table it is much easier to decode since the letters in the encoded word are now in alphabetical order in the top line.

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

These simple substitution ciphers are fairly easy to "crack"—the problem is that in English (or any language), certain letters are far more likely to appear. In English, for example, the letter "E" is far more likely to appear than the letter "Z". In fact, here is a list of the letters used in English arranged approximately in order of usage ("E" is the most used letter; "Z" is least). The approximate percentages for the first few letters in the list below are: E: 12.7%, T: 9.1%, A: 8.2%, O: 7.5%, and the percentages for the last few are: J: 0.2%, Q: 0.1%, Z: 0.1%.

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

Following is a short passage encoded with a simple substitution cipher:

```
  UJEJVZR QFEYGE, SV SO OFSU, JWIG FEESTGU FV VZG
UJJE JC FW FQFEVLGWV SW PZSIZ F NASVVGESWN QFEVR
PFO VFYSWN QAFIG.  FV QEGISOGAR VZG OFLG LJLGWV,
F XGFKVSCKA XKV TFIKJKO OZJPNSEA FEESTGU FV VZG
UJJE.
  CJE F LJLGWV, VZGEG PFO ZGOSVFVSJW JW XJVZ OSUGO,
FWU VZGW VZG OZJPNSEA OVGQQGU XFIY VJ LFYG PFR,
OFRSWN, "FNG XGCJEG XGFKVR."
  "WJV FV FAA!" OFSU UJEJVZR QFEYGE, OFSASWN
VZEJKNZ. "QGFEAO XGCJEG OPSWG!"
```

To try to crack this cipher, begin by counting the number of occurrences of each letter, and we obtain the following counts:

A	B	C	D	E	F	G	H	I	J	K	L	M
10	0	5	0	24	35	34	0	6	24	7	7	0
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
9	18	7	9	7	22	3	11	31	16	7	5	16

Since the text sample was relatively small, we can't be certain that the most common letter ("F" in the sample above) stands for the letter "E", but it's a pretty good bet that you'll find "E" among the letters "F", "G", and "V".

The structure of English gives plenty of other clues as well. For example, the word "F" appears twice in the text, so "F" must stand for "I" or "A". Since "F" is very common in the sample above, it is more likely to stand for "A", since "A" is much more common in English than "I". So the first guess you might make is that "F" stands for "A". Now the word "FV" appears in the text twice, and "V" is also very common. "AT" is a word in English, so perhaps "V" stands for "T" in the cipher. Now these are just guesses, but they are not bad guesses.

Making those substitutions gives us the following:

```
      T    A       T     A         A       AT T
  UJEJVZR QFEYGE, SV SO OFSU, JWIG FEESTGU FV VZG
        A  A A T   T          A    TT       A T
UJJE JC FW FQFEVLGWV SW PZSIZ F NASVVGESWN QFEVR
 A  TA       A     AT           T    A        T
PFO VFYSWN QAFIG.  FV QEGISOGAR VZG OFLG LJLGWV,
A   A T       T  A               A       AT T
F XGFKVSCKA XKV TFIKJKO OZJPNSEA FEESTGU FV VZG

UJJE.
      A      T  T      A      TAT         T
  CJE F LJLGWV, VZGEG PFO ZGOSVFVSJW JW XJVZ OSUGO,
A   T    T             T       A   T   A    A
FWU VZGW VZG OZJPNSEA OVGQQGU XFIY VJ LFYG PFR,
 A       A              T
OFRSWN, "FNG XGCJEG XGFKVR."
     T AT A      A       T    A       A
  "WJV FV FAA!" OFSU UJEJVZR QFEYGE, OFSASWN
T           A
VZEJKNZ. "QGFEAO XGCJEG OPSWG!"
```

Looking at the text above, there are a lot more clues. For one thing, in the first line is the word “SV”, where the “V” may stand for “T”. The only two words in English ending in “T” are “AT” and “IT”, but we’ve already guessed that “F” stands for “A”, so “S” is probably “I”. Also, since we think we know what letters stand for “T” and “A”, the other extremely common letter, “G”, probably stands for “E”. Finally, in the next-to-last line is the word “FAA”—a three letter word beginning with “A”. In English, “A” must be “L”, “D”, or “S”, but “at all” makes much more sense than “at add” or “at ass”, so “A” is probably “L”:

```
      T    A  E   IT I   AI      E A  I E  AT T E
  UJEJVZR QFEYGE, SV SO OFSU, JWIG FEESTGU FV VZG
        A  A A T E T I    I   A  LITTE I    A T
UJJE JC FW FQFEVLGWV SW PZSIZ F NASVVGESWN QFEVR
 A  TA I    LA E   AT   E I EL  T E  A E    E T
PFO VFYSWN QAFIG.  FV QEGISOGAR VZG OFLG LJLGWV,
A  EA TI  L   T  A           I L A  I E  AT T E
F XGFKVSCKA XKV TFIKJKO OZJPNSEA FEESTGU FV VZG

UJJE.
      A    E T  T E E  A   E ITATI        T   I E
  CJE F LJLGWV, VZGEG PFO ZGOSVFVSJW JW XJVZ OSUGO,
A   T E  T E      I L  TE  E   A   T   A E  A
FWU VZGW VZG OZJPNSEA OVGQQGU XFIY VJ LFYG PFR,
 A I     A E  E   E  EA T
OFRSWN, "FNG XGCJEG XGFKVR."
     T AT ALL    AI      T    A  E    AILI
  "WJV FV FAA!" OFSU UJEJVZR QFEYGE, OFSASWN
T          EA L   E   E   I E
VZEJKNZ. "QGFEAO XGCJEG OPSWG!"
```

From here, it’s easy to make progress. In the next-to-last line, “WJV FV FAA” is almost certainly “NOT AT ALL”, so “W” is “N” and “J” is “O”. Similarly, the word “VZG” is almost certainly “THE”, so “Z” is “H”. Thus we obtain:

```
   O OTH   A  E   IT I   AI   ON E A  I E  AT THE
  UJEJVZR QFEYGE, SV SO OFSU, JWIG FEESTGU FV VZG
 OO  O  AN A A T ENT IN  HI H A  LITTE IN   A T
UJJE JC FW FQFEVLGWV SW PZSIZ F NASVVGESWN QFEVR
 A  TA IN   LA E   AT   E I EL  THE  A E  O ENT
PFO VFYSWN QAFIG.  FV QEGISOGAR VZG OFLG LJLGWV,
A  EA TI  L   T  A  O    HO   I L A  I E  AT THE
F XGFKVSCKA XKV TFIKJKO OZJPNSEA FEESTGU FV VZG
 OO
UJJE.
   O  A  O ENT  THE E  A  HE ITATION ON  OTH  I E
  CJE F LJLGWV, VZGEG PFO ZGOSVFVSJW JW XJVZ OSUGO,
AN  THEN THE  HO I L  TE  E   A   TO  A E  A
FWU VZGW VZG OZJPNSEA OVGQQGU XFIY VJ LFYG PFR,
 A IN    A E  E O E  EA T
OFRSWN, "FNG XGCJEG XGFKVR."
   NOT AT ALL    AI   O OTH   A  E    AILIN
  "WJV FV FAA!" OFSU UJEJVZR QFEYGE, OFSASWN
TH O  H    EA L   E O E   INE
```

```
VZEJKNZ. "QGFEAO XGCJEG OPSWG!"
```

From what we have above, “FWU” is clearly “AND”, so “U” codes for “D”, “ZGOSVFVSJW” is “HESITATION”, so “O” codes for “S”, “VZGEG” is either “THESE” or “THERE”, but “S” is used, so “E” codes for “R”:

```
  DOROTH   AR ER  IT IS SAID  ON E ARRI ED AT THE
  UJEJVZR QFEYGE, SV SO OFSU, JWIG FEESTGU FV VZG
DOOR O  AN A ART ENT IN  HI H A  LITTERIN   ART
UJJE JC FW FQFEVLGWV SW PZSIZ F NASVVGESWN QFEVR
 AS TA IN   LA E   AT  RE ISEL  THE SA E  O ENT
PFO VFYSWN QAFIG.  FV QEGISOGAR VZG OFLG LJLGWV,
A  EA TI  L   T  A  O   SHO  IRL ARRI ED AT THE
F XGFKVSCKA XKV TFIKJKO OZJPNSEA FEESTGU FV VZG
DOOR
UJJE.
   OR A  O ENT  THERE  AS HESITATION ON  OTH SIDES
  CJE F LJLGWV, VZGEG PFO ZGOSVFVSJW JW XJVZ OSUGO,
AND THEN THE SHO  IRL STE  ED  A   TO  A E  A
FWU VZGW VZG OZJPNSEA OVGQQGU XFIY VJ LFYG PFR,
SA IN    A E  E ORE  EA T
OFRSWN, "FNG XGCJEG XGFKVR."
   NOT AT ALL   SAID DOROTH   AR ER  SAILIN
  "WJV FV FAA!" OFSU UJEJVZR QFEYGE, OFSASWN
THRO  H    EARLS  E ORE S INE
VZEJKNZ. "QGFEAO XGCJEG OPSWG!"
```

From here, it’s easy. Fill in the obvious letters for a couple of passes to obtain the final decryption:

```
  DOROTHY PARKER, IT IS SAID, ONCE ARRIVED AT THE
  UJEJVZR QFEYGE, SV SO OFSU, JWIG FEESTGU FV VZG
DOOR OF AN APARTMENT IN WHICH A GLITTERING PARTY
UJJE JC FW FQFEVLGWV SW PZSIZ F NASVVGESWN QFEVR
WAS TAKING PLACE.  AT PRECISELY THE SAME MOMENT,
PFO VFYSWN QAFIG.  FV QEGISOGAR VZG OFLG LJLGWV,
A BEAUTIFUL BUT VACUOUS SHOWGIRL ARRIVED AT THE
F XGFKVSCKA XKV TFIKJKO OZJPNSEA FEESTGU FV VZG
DOOR.
UJJE.
  FOR A MOMENT, THERE WAS HESITATION ON BOTH SIDES,
  CJE F LJLGWV, VZGEG PFO ZGOSVFVSJW JW XJVZ OSUGO,
AND THEN THE SHOWGIRL STEPPED BACK TO MAKE WAY,
FWU VZGW VZG OZJPNSEA OVGQQGU XFIY VJ LFYG PFR,
SAYING, "AGE BEFORE BEAUTY."
OFRSWN, "FNG XGCJEG XGFKVR."
  "NOT AT ALL!" SAID DOROTHY PARKER, SAILING
  "WJV FV FAA!" OFSU UJEJVZR QFEYGE, OFSASWN
THROUGH. "PEARLS BEFORE SWINE!"
VZEJKNZ. "QGFEAO XGCJEG OPSWG!"
```

2.1 Improving a Substitution Cipher

There are plenty of ways to improve on the simplest form of substitution cipher as presented above. This section lists some obvious (and not so obvious) things you can do.

Most important, get rid of the spaces, punctuation, et cetera, or at the very least, encode them with alternative symbols as well. Just an encoding of the spaces won't help much—there are vastly more spaces in any document than any other character, so the one that stands for a space will be obvious.

But you can fix that problem (and the frequency problem in general) by having duplicate ciphers. For example, don't restrict yourself to just 26 output symbols. (For example, imagine that your encoding will consist of integers between 0 and 999.) Then instead of having one number stand for "E", have a whole bunch of them stand for it so that the frequencies balance out a bit. Using the frequencies of letters in English we stated earlier (E: 12.7%, T: 9.1%, A: 8.2%, O: 7.5%, . . . , J: 0.2%, Q: 0.1%, Z: 0.1%), we'd have 127 different symbols for "E", 91 for "T", and 1 for "Q" and for "Z". If we do this, the person trying to break the code will find that all the symbols will be approximately equally common, and it will be much harder to get started.

But such a code is still subject to an attack on frequencies. For example, if the codebreaker has access to a lot of text, letter pair frequencies can be used. For example, in English, whenever there is a "Q" in the text, you can be virtually certain that it will be followed by a "U" (except if you're talking about Iraq). Similarly "TH" and "EN" are very common, and you hardly ever see "BD" (bdelium). So even an augmented cipher as described in the previous paragraph can be attacked with a frequency analysis.

Of course there's nothing to prevent the code maker from having encodings for letter pairs, triplets, common words, et cetera to make the task still more difficult. The disadvantage is that as you add more combinations, the ciphers become more and more difficult to decode.

Another option along the same lines is simply to add "nulls" to the cipher. Encodings that are just garbage and should be tossed out during the decoding. These can be used to balance frequencies in a nice way.

And of course you can add special items that mean things like "ignore the next item", or "delete the previous item". But in spite of all the suggestions above, if you have a sufficient amount of text encoded with a simple substitution cipher, it is only a matter of time before someone could break it.

Here's an example of a straight substitution cipher that you can try to break:

```
  NOT NUA JMPETZ UTST ZENNELV EL NOT

JEGELV SAAW, UMENELV KAS NOTES

OAZNTZZ, UOA UMZ ZJEVONJX PTJMXTP.

NOT PMCVONTS AK NOT KMWEJX UMZ UENO

NOTW, AL NOT NOTASX NOMN ZOT UACJP

QTTY NOT GEZENASZ ADDCYETP PCSELV

NOT UMEN.

  NOT DOEJP UMZ YTSOMYZ ZEI XTMSZ

AJP, ZLCR LAZTP, KSTDQTJP, RCDQ
```

```
NAANOTP MLP RTZYTDNMDJTP.  ZOT

WMELNMELTP M PTTY ZEJTLDT MLP

NOT NUA JMPETZ YTTSTP PACRNKCJJX

MN OTS.

  KELMJJX, ALT AK NOTW WCNNTSTP

NA NOT ANOTS, "LAN GTSX Y-S-T-N-N-X,

E KTMS," DMSTKCJJX ZYTJJELV NOT QTX

UASP.

  UOTSTCYAL NOT DOEJP YEYTP CY,

"RCN MUKCJ Z-W-M-S-N!"
```

To save you some time, here are the character frequencies for the passage above:

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
25	0	14	10	28	0	3	0	1	23	10	18	25	44	31	24	4	5	22	60	14	7	7	11	13	23

The answer appears in the final section of this document.

3 Permutations

Another obvious method to encode a message is to jumble the letters in some way so that the recipient can simply unjumble them. Here's a very simple way to do it that's not very secure, but it shows you the idea.

First, break the message up into chunks of 25 letters. If there is some left over at the end, add enough junk to extend the last chunk to be 25 letters, too. Each of the chunks is then encoded by itself. Here's an example of the encoding of a very short message: "WHO IS THAT CUTE GIRL?".

First off, it is not even 25 characters long, so let's extend it (I'll use the letter "X" for the extension, but it would be much better to use more innocent-looking letters. Here's the 25 character chunk: "WHOISTHATCUTEGIRL?XXXXXXX".

Next, lay out the letters in a 5×5 square as follows (obviously it would be better to leave out the the punctuation):

W	H	O	I	S
T	H	A	T	C
U	T	E	G	I
R	L	?	X	X
X	X	X	X	X

Then just read out the text by columns instead of rows, so the encoding of the phrase becomes: "WTURXHHTLXOAE?XITGXXSCIXX". To decode the message, simply write characters in the columns of a 5×5 grid and read the message out of the rows.

If this is the only thing that's been done, it is very easy to crack the encoding. For one thing, when you check the frequency, you notice that it is the same as in standard English, so it may just be a transposition code. But there is nothing to prevent you from doing a substitution cipher first, and then encoding as above. The nice thing about the above jumbling is that it will destroy any letter pair or letter triple frequencies.

4 Vigenère Cipher

A Vigenére Cipher is just a mixture of the 26 different ciphers shown in the table below:

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

The 26 ciphers are called "A", "B", et cetera, and are listed in the first column. If you wish to look up the letter "G" in the cipher "C", go to the third row (labelled "C") and look up the letter under the "G". The result is "I". The encoding of the word "DILBERT" using the "F" cipher is "INQGJWY".

The Vigenère cipher uses different individual ciphers for each letter. The usual method is to agree upon a keyword with the person you wish to receive the message. Let's use "FROGLEGS" as our keyword for this example. Now suppose you wish to encode the phrase "ONCE UPON A MIDNIGHT DREARY".

Begin by writing "FROGLEGS" repeatedly over the phrase:

```
F R O G  L E G S  F  R O G L E G S F  R O G L E G
O N C E  U P O N  A  M I D N I G H T  D R E A R Y
```

To perform the encoding, we use the "F" cipher on the "O" of "ONCE", the "R" code on the "N", the "O" code on the "C", and so on, yielding:

TEQK FTUF F DWJYMMZY UFKLVE

To make sure you understand what is going on, decipher the following phrase using the Vigenère cipher with the keyword CRYPTOGRAPHY:

HFSGLQUIE PUB UVTTG MKRRH HEQ

Cracking a code like this is a bit more difficult than cracking a simple substitution code, but it is certainly possible. The longer the keyword, of course, the more difficult it is to crack the code. If the keyword were infinitely long, and random, it would be impossible to decode, since any decoding is as likely as any others. But when this sort of code is used, typically the keyword is a real English word or phrase so that it is not too hard to remember.

A very long keyword of random letters is unbreakable, and here's why. Any coding can represent any text with some keyword. For example, suppose the message is "WEASEL". With an appropriate "keyword", it can represent anything that's 6 characters long. Suppose you want "WEASEL" to represent "TURNIP"—just use the table and figure out what the right keyword to do this would be. "W" maps to "T", so the first letter of the keyword is "X". "E" maps to "U", so the second letter of the keyword has to be "Q", and so on. Continuing in this way, we discover the keyword is "XQRVEE". Similarly, if the keyword were "HNAOKC", then "WEASEL" represents "DRAGON".

Clearly, if the key is allowed to be arbitrarily long and composed of arbitrary letters, then anything can stand for anything, and hence the code is completely secure. But the first time you reuse the key, you give the code breaker some information that can help. If the key is 1000 characters long, if you sent a message of a million characters, the key would have to be reused 1000 times, and it would not be at all secure.

5 Converting Text to Numbers

Some of the more mathematical methods of encryption that we'll discuss are best described in terms of transformations of integers into other integers. For the more interesting encodings, it really doesn't matter how you do this translation, but certain methods have slight advantages over others.

The most common method is probably the ASCII encoding that assigns a number to each character. The standard ASCII encoding specifies 128 different characters (including not only the standard upper and lower case letters and the digits, but all the punctuation, some control characters, and various other things.

Here is the standard ASCII encoding. The numbers on the sides and top are in octal (base 8) and need to be combined. For example, the character "S" is in row 12, column 3. It's ASCII octal number is thus 123. To convert to base 10, 123 (octal) $= 1 \cdot 8^2 + 2 \cdot 8^1 + 3 \cdot 8^0 = 64 + 16 + 3 =$ 83 (decimal). 040 (octal) is the space character; 177 (octal) is the "delete" character.

	0	1	2	3	4	5	6	7
00	ˆ@	ˆA	ˆB	ˆC	ˆD	ˆE	ˆF	ˆG
01	ˆH	ˆI	ˆJ	ˆK	ˆL	ˆM	ˆN	ˆO
02	ˆP	ˆQ	ˆR	ˆS	ˆT	ˆU	ˆV	ˆW
03	ˆX	ˆY	ˆZ	ˆ[	ˆ\	ˆ]	ˆˆ	ˆ_
04		!	”	#	$	%	&	’
05	(	)	*	+	,	-	.	/
06	0	1	2	3	4	5	6	7
07	8	9	:	;	<	=	>	?
10	@	A	B	C	D	E	F	G
11	H	I	J	K	L	M	N	O
12	P	Q	R	S	T	U	V	W
13	X	Y	Z	[	\	]	ˆ	_
14	‘	a	b	c	d	e	f	g
15	h	i	j	k	l	m	n	o
16	p	q	r	s	t	u	v	w
17	x	y	z	{	|	}	∼	DEL

The ASCII assignments are for the numbers from 0 to 127, which require 7 bits of data. The standard character on a computer (one byte) is 8 bits of data, which can represent a number from 0 to 255. When ASCII encoding is used, each character is put in one byte, so effectively, one bit of each character is wasted.

But the nice thing about a numeric representation that packs into 7 or 8 bits is that the numeric representations can simply be concatenated to make representations of groups of letters. If you wish to encode two ASCII characters at once, simply place their binary representations next to each other, and you get a 16 bit number (between 0 and 65535). If you don’t understand this, imagine that you are working in base 10, and you have a character set that encodes to a number between 00 and 99. Then if “A” happened to be 17 and “Z” were 42, then the character pair “AZ” would be the four-digit number 1742.

Binary representations work great on a computer, but since many people find base 10 much easier to work with than base 2, the numeric examples in this paper will use the following encoding:

	0	1	2	3	4	5	6	7	8	9
0	XX	XX	XX	XX	XX	XX	XX	XX	XX	XX
1	SP	A	B	C	D	E	F	G	H	I
2	J	K	L	M	N	O	P	Q	R	S
3	T	U	V	W	X	Y	Z	a	b	c
4	d	e	f	g	h	i	j	k	l	m
5	n	o	p	q	r	s	t	u	v	w
6	x	y	z	.	,	:	;	’	“	‘
7	!	@	#	$	%	ˆ	&	*	-	+
8	(	)	[	]	{	}	?	/	<	>
9	0	1	2	3	4	5	6	7	8	9

The entries with XX are not used, and “SP” is the space character. Thus all characters code to between 10 and 99. “U” is 31; “5” is 95, and so on. If we want to encode four characters at a time, for example, we would encode the word “C3PO” as the 8-digit number 13932625.

6 Generating Long Keys

What you would like to have is a very long key to use in a scheme like the Vigenére encoding, but as we stated above, the problem with a long key is that you have to transmit that key to the persons who should be able to decode the messages in some secure manner.

The more “random” the key is, the more secure it is. To make a really random key, you could imagine using something like a timer connected to a Geiger counter and placing it close to some radioactive source so you could use the intervals between atomic decays to generate numbers that are truly random.

But if you’re willing just to have a sequence that looks very random, there are many techniques for generating such sequences of so-called “pseudo-random numbers”. Volume 2 of Donald Knuth’s classic book, “The Art of Computer Programming—Seminumerical Algorithms”, contains a treasure-trove of information about how to generate pseudo-random sequences. We will look at just a couple of methods here to give you an idea of the possibilities.

A very easy way to do this (but not a very good one) is based on the fact that if p is a prime number, and n is any number with $0 < n < p$, then the sequence of numbers $kn (\bmod p)$ cycle through all the numbers between 1 and $p-1$ without missing any of them, and in an order that appears somewhat random as k goes through $1, 2, 3, \ldots, p-1$.

For example, if $p = 13$ and $n = 5$:

$$\begin{array}{ccc}
5 \cdot 1 (\bmod 13) = 5 & 5 \cdot 2 (\bmod 13) = 10 & 5 \cdot 3 (\bmod 13) = 2 \\
5 \cdot 4 (\bmod 13) = 7 & 5 \cdot 5 (\bmod 13) = 12 & 5 \cdot 6 (\bmod 13) = 4 \\
5 \cdot 7 (\bmod 13) = 9 & 5 \cdot 8 (\bmod 13) = 1 & 5 \cdot 9 (\bmod 13) = 6 \\
5 \cdot 10 (\bmod 13) = 11 & 5 \cdot 11 (\bmod 13) = 3 & 5 \cdot 12 (\bmod 13) = 8
\end{array}$$

The numbers cycle through the series: $5, 10, 2, 7, 12, 4, 9, 1, 6, 11, 3, 8$. This sequence has the vague appearence of being random. If you choose a larger prime, the sequence will be longer.

Using such a sequence, let’s encrypt a message using the code described in the previous section. The message will be:

PASSWORD: ELEPHANT

Including the space, the numeric codes for these letters are:

26, 11, 29, 29, 33, 25, 28, 14, 65, 10, 15, 22, 15, 26, 18, 11, 24, 30

Now we’ll use a combination of the method above to generate a sequence of pseudo-random numbers to use as a sort of Vigenére key. We’ll let $p = 16139$ and $n = 4352$. As k goes from 1 to 18 (there are 18 characters in the message above), $kn (\bmod 16139)$ goes through the following 18 numbers:

4352, 8704, 13056, 1269, 5621, 9973, 14325, 2538, 6890, 11242, 15594, 3807, 8159, 12511, 724, 5076, 9428, 13780

We will just encode our numbers by adding these pseudo-random numbers to the numbers in our sequence and taking the result modulo 100. For example, the first letter, 26, is converted to (26+4352) $(\bmod 100)$ = 78. The encoded sequence becomes:

78, 15, 85, 98, 54, 98, 53, 52, 55, 52, 09, 29, 74, 37, 42, 87, 52, 10

Assuming that the person who wishes to decode the message knows the values of p and n, he can generate the exact same sequence of keys, and can undo the encryption above. For example, to decrypt the first character, he generates the first number in the key sequence, 4352, subtracts that from 78 (giving -4274), and taking that result modulo 100 to get 26.

Now notice that the only information that has to be passed to the recipient as the key is the pair of numbers p and n—there is no need to transmit a long sequence of keys.

6.1 Better Pseudo-Random Sequences

The sequence above is too simple, but with just a tiny modification, it can be made much more interesting. The following sequence can work quite well.

Choose a (usually large) prime number p and two integers a and c. In addition, choose a “seed”, or starting number for the random sequence which we will call X_0 such that $0 <= X_0 < p$. Here is the formula for X_i, if $i > 0$:

$$X_i = (aX_{i-1} + c)(\bmod p).$$

So, for example, if $p = 16139$, $a = 91$, and $c = 541$, and we begin with $X_0 = 11111$, we generate the following sequence:

$$\begin{aligned}
X_0 &= 11111\\
X_1 &= (91 \cdot 11111 + 541) (\text{mod } 16139) = 11024\\
X_2 &= (91 \cdot 11024 + 541) (\text{mod } 16139) = 3107\\
X_3 &= (91 \cdot 3107 + 541) (\text{mod } 16139) = 8915\\
X_4 &= (91 \cdot 8915 + 541) (\text{mod } 16139) = 4856\\
X_5 &= (91 \cdot 4856 + 541) (\text{mod } 16139) = 6684
\end{aligned}$$

which you can continue as long as you want. The first pseudo-random numbers in this sequence are: 11111, 11024, 3107, 8915, 4856, 6684. Of course the sequence has to cycle in p or fewer steps.

As an exercise, try to decode the following message based on $p = 16139$, $a = 91$, and $c = 541$, but with the starting value $X_0 = 0$. We will use the random sequence in the same way we did above—we converted our message to numbers between 10 and 99 using the table in Section 5, then we added successive keys and took the result modulo 100. Here is the resulting encoded message:

23, 52, 85, 91, 15, 06, 53, 61, 30, 72, 23

To get you started, the first number in the sequence is zero, so $X_0 = 0$, so the first number decodes as $(23 - 0) (\text{mod } 100) = 23$, so the first letter in the message is "M". Then

$$X_1 = (91 \cdot X_0 + 541) (\text{mod } 16139) = 541 (\text{mod } 16139) = 541,$$

so the second letter is $(52 - 541) (\text{mod } 100) = 11$, so the second letter is "A". Continue in the same way.

7 The German "Enigma" Code

During the second world war, the German military used a special encrytion machine called "Enigma" to encode its messages. Basically, the Enigma machine generated, given a "seed", a sequence of numbers that appeared sufficiently random to make them very difficult to crack, even if the internal details of the machine were known.

The allies had captured some of the Enigma machines, so they did know the internal workings, but much of the foundations of modern computer science were developed in an attempt to crack the German messages (usually with a fair amount of success).

It's beyond the scope of this paper to describe the internals of the Enigma machine and encoding technique, but here are a couple of references. See "The Code Book", by Simon Singh, or take a look at the web site:

`http://www.gl.umbc.edu/ lmazia1/Enigma/enigma.html`

8 Public Key Cryptography

One of the biggest problems in cryptography is the distribution of keys. Suppose you live in the United States and want to pass information secretly to your friend in Europe. If you truly want to keep the information secret, you need to agree on some sort of key that you and he can use to encode/decode messages. But you don't want to keep using the same key, or you will make it easier and easier for others to crack your cipher.

But it's also a pain to get keys to your friend. If you mail them, they might be stolen. If you send them cryptographically, and someone has broken your code, that person will also have the next key. If you have to go to Europe regularly to hand-deliver the next key, that is also expensive. If you hire some courier to deliver the new key, you have to trust the courier, et cetera.

8.1 Trap-Door Ciphers

But imagine the following situation. Suppose you have a special method of encoding and decoding that is "one way" in a sense. Imagine that the encoding is easy to do, but decoding is very difficult. Then anyone in the world can encode a message, but only one person can decode it. Such methods exist, and they are called "one way ciphers" or "trap door ciphers".

Here's how they work. For each cipher, there is a key for encoding and a different key for decoding. If you know the key for decoding, it is very easy to make the key for encoding, but it is almost impossible to do the opposite—to start with the encoding key and work out the decoding key.

So to communicate with your friend in Europe, each of you has a trap door cipher. You make up a decoding key D_a and generate the corresponding encoding key E_a. Your friend does exactly the same thing, but he makes up a decoding key D_b and generates the corresponding encoding key E_b. You tell him E_a (but not D_a) and he tells you E_b (but not D_b). Then you can send him messages by encoding using E_b (which only he can decode) and vice-versa—he encodes messages to you using E_a (which only you can decode, since you're the only person with access to D_a).

Now if you want to change to a new key, it is no big problem. Just make up new pairs and exchange the encoding keys. If the encoding keys are stolen, it's not a big deal. The person who steals them can only encode messages—they can't decode them. In fact, the encoding keys (sometimes called "public keys") could just be published in a well-known location. It's like saying, "If you want to send me a private message, encode it using this key, and I will be the only person in the world who can read it." But be sure to keep the decoding key (the "private key") secret.

8.2 Certification

There is, of couse, a problem with the scheme above. Since the public keys are really public, anyone can "forge" a message to you. So your enemy can pretend to be your friend and send you a message just like your friend can—they both have access to the public key. Your enemy's information can completely mislead you. So how can you be certain that a message that says it is from your friend is really from your friend?

Here is one way to do it, assuming that you both have the public and private keys E_a, E_b, D_a, and D_b as discussed in the previous section. Suppose I wish to send my friend a message that only he can read, but in such a way that he is certain that the message is from me. Here's how to do it.

I will take my name, and pretend that it is an encoded message, and decode it using D_a. I am the only person who can do this, since I am the only person who knows D_a. Then I include that text in the real message I wish to send, and I encode the whole mess using E_b, which only my friend knows how to decode.

When he receives it, he will decode it using D_b, and he will have a message with an additional piece of what looks to him like junk characters. The junk characters are what I got by "decoding" my name. So he simply encodes the junk using my public key E_a and makes certain that it is my name. Since I am the only one who knows how to make text that will encode to my name, he knows the message is from me.

You can encode any text for certification, and in fact, you should probably change it with each message, but it's easy to do. Your message to your friend would look like this:

"Attack at dawn. Here is my decoding of 'ABCDEFG': 'JDLEODK'."

To assure privacy, for each message, change the "ABCDEFG" and the corresponding "JDLEODK".

9 RSA Encryption

OK, in the previous section we described what is meant by a trap-door cipher, but how do you make one? One commonly used cipher of this form is called "RSA Encryption", where "RSA" are the initials of the three creators: "Rivest, Shamir, and Adleman". It is based on the following idea:

It is very simply to multiply numbers together, especially with computers. But it can be very difficult to factor numbers. For example, if I ask you to multiply together 34537 and 99991, it is a simple matter to punch those numbers into a calculator and 3453389167. But the reverse problem is much harder.

Suppose I give you the number 1459160519. I'll even tell you that I got it by multiplying together two integers. Can you tell me what they are? This is a very difficult problem. A computer can factor that number fairly quickly, but (although there are some tricks) it basically does it by trying most of the possible combinations. For any size number, the computer has to check something that is of the order of the size of the square-root of the number to be factored. In this case, that square-root is roughly 38000.

Now it doesn't take a computer long to try out 38000 possibilities, but what if the number to be factored is not ten digits, but rather 400 digits? The square-root of a number with 400 digits is a number with 200 digits. The lifetime of the universe is approximately 10^{18} seconds – an 18 digit number. Assuming a computer could test one million factorizations per second, in the lifetime of the universe it could check 10^{24} possibilities. But for a 400 digit product, there are 10^{200} possibilties. This means the computer would have to run for 10^{176} times the life of the universe to factor the large number.

It is, however, not too hard to check to see if a number is prime—in other words to check to see that it cannot be factored. If it is not prime, it is difficult to factor, but if it is prime, it is not hard to show it is prime.

So RSA encryption works like this. I will find two huge prime numbers, p and q that have 100 or maybe 200 digits each. I will keep those two numbers secret (they are my private key), and I will multiply them together to make a number $N = pq$. That number N is basically my public key. It is relatively easy for me to get N; I just need to multiply my two numbers. But if you know N, it is basically impossible for you to find p and q. To get them, you need to factor N, which seems to be an incredibly difficult problem.

But exactly how is N used to encode a message, and how are p and q used to decode it? Below is presented a complete example, but I will use tiny prime numbers so it is easy to follow the arithmetic. In a real RSA encryption system, keep in mind that the prime numbers are huge.

In the following example, suppose that person A wants to make a public key, and that person B wants to use that key to send A a message. In this example, we will suppose that the message A sends to B is just a number. We assume that A and B have agreed on a method to encode text as numbers as was discussed in Section 5. Here are the steps:

1. Person A selects two prime numbers. We will use $p = 23$ and $q = 41$ for this example, but keep in mind that the real numbers person A should use should be *much* larger.

2. Person A multiplies p and q together to get $pq = (23)(41) = 943$. 943 is the "public key", which he tells to person B (and to the rest of the world, if he wishes).

3. Person A also chooses another number e which must be relatively prime to $(p - 1)(q - 1)$. In this case, $(p - 1)(q - 1) = (22)(40) = 880$, so $e = 7$ is fine. e is also part of the public key, so B also is told the value of e.

4. Now B knows enough to encode a message to A. Suppose, for this example, that the message is the number $M = 35$.

5. B calculates the value of $C = M^e (\mathrm{mod}\ N) = 35^7 (\mathrm{mod}\ 943)$.

6. $35^7 = 64339296875$ and $64339296875 (\mathrm{mod}\ 943) = 545$. The number 545 is the encoding that B sends to A.

7. Now A wants to decode 545. To do so, he needs to find a number d such that $ed = 1(\mathrm{mod}\ (p - 1)(q - 1))$, or in this case, such that $7d = 1(\mathrm{mod}\ 880)$. A solution is $d = 503$, since $7 * 503 = 3521 = 4(880) + 1 = 1(\mathrm{mod}\ 880)$.

8. To find the decoding, A must calculate $C^d (\mathrm{mod}\ N) = 545^{503} (\mathrm{mod}\ 943)$. This looks like it will be a horrible calculation, and at first it seems like it is, but notice that $503 = 256 + 128 + 64 + 32 + 16 + 4 + 2 + 1$ (this is

just the binary expansion of 503). So this means that

$$545^{503} = 545^{256+128+64+32+16+4+2+1} = 545^{256}545^{128}\cdots 545^{1}.$$

But since we only care about the result $\pmod{943}$, we can calculate all the partial results in that modulus, and by repeated squaring of 545, we can get all the exponents that are powers of 2. For example, $545^2 \pmod{943} = 545 \cdot 545 = 297025 \pmod{943} = 923$. Then square again: $545^4 \pmod{943} = (545^2)^2 \pmod{943} = 923 \cdot 923 = 851929 \pmod{943} = 400$, and so on. We obtain the following table:

$$\begin{aligned}
545^{1} \pmod{943} &= 545\\
545^{2} \pmod{943} &= 923\\
545^{4} \pmod{943} &= 400\\
545^{8} \pmod{943} &= 633\\
545^{16} \pmod{943} &= 857\\
545^{32} \pmod{943} &= 795\\
545^{64} \pmod{943} &= 215\\
545^{128} \pmod{943} &= 18\\
545^{256} \pmod{943} &= 324
\end{aligned}$$

So the result we want is:

$$545^{503} \pmod{943} = 324 \cdot 18 \cdot 215 \cdot 795 \cdot 857 \cdot 400 \cdot 923 \cdot 545 \pmod{943} = 35.$$

Using this tedious (but simple for a computer) calculation, A can decode B’s message and obtain the original message $N = 35$.

9.1 RSA Exercise

OK, now to see if you understand the RSA decryption algorithm, suppose you are person A, and you have chosen as your two primes $p = 97$ and $q = 173$, and you have chosen $e = 5$. Thus you told B that $N = 16781$ (which is just pq) and you told him that $e = 5$.

He encodes a message (a number) for you and tells you that the encoding is 5347. Can you figure out the original message?

The answer appears in the Solutions section below.

10 Solutions

The solution to the cipher is this:

```
  THE TWO LADIES WERE SITTING IN THE
LIVING ROOM WAITING FOR THEIR
HOSTESS, WHO WAS SLIGHTLY DELAYED.
THE DAUGHTER OF THE FAMILY WAS WITH
THEM, ON THE THEORY THAT SHE WOULD
KEEP THE VISITORS OCCUPIED DURING THE WAIT.
```

```
  THE CHILD WAS PERHAPS SIX YEARS
OLD, SNUB NOSED, FRECKELD, BUCK
TOOTHED AND BESPECTACLED.  SHE
MAINTAINED A DEEP SILENCE AND
THE TWO LADIES PEERED DOUBTFULLY
AT HER.
  FINALLY, ONE OF THEM MUTTERED
TO THE OTHER, "NOT VERY P-R-E-T-T-Y,
I FEAR," CAREFULLY SPELLING THE KEY
WORD.
  WHEREUPON THE CHILD PIPED UP,
"BUT AWFUL S-M-A-R-T!"
```

The answer for the RSA exercise is 16657