Part IV
Transposition ciphers

Unit 51
Transposition ciphers

A *transposition cipher* is a cipher that rearranges the letters of a text, but does no substitutions.

Detecting a transposition cipher can be done by looking for a high monogram fitness but a low tetragram fitness. The index of coincidence should be close to that of English, since IoC does not depend on the order of letters. Note that for short texts, we cannot rely too much on statistics.

There is a type of transposition cipher called a *route transposition*. It involves choosing some special path through the text and adding letters to the ciphertext as we traverse that path. Since it does not typically use a mathematical system for choosing that path, we will not be discussing it any further in this book.

Reading and references

Helen Fouché Gaines, *Cryptanalysis: a study of ciphers and their solution*, New York: Dover, 1956; previously titled *Elementary Cryptanalysis* and published by American Photographic in 1939; archive.org/details/cryptanalysis00gain; chapter III.

Programming tasks

1. Write a function to return a boolean value representing whether a given ciphertext is likely to have been encrypted with a transposition cipher.

Exercises

1. Which of these are encrypted with transposition ciphers?

    a. `IENEEMEEMIENYNEICOEMHTCAGTIAYRBETHETFEIOOEHHRLELTLESGIMH NEEOEEMIENIEEINMEMOY`

    b. `EEISEHEEISEHSIEYHJENDQNUDQSOEMAYQUEQJESRUEUJFFEMPFEQUSHO JEEISEHEEISEHSIEYHJE`

c. JMSTUYTWIBCZLCBNHIPWIBPESIUNNIQDSRSZYLDDFITCLTULTCZDPIBG
DPCZLCBNHRWMNYIDSWLTUIWMBTCSWMQEMQHYIYRYTWIBCIDMQJDAJFCO
MEDMQWIHPHODPIBIDSLCSHSYSTTPIBCIDMQYMEBHISWOBMESIDSVYTRN
PHHDPIOPIYSTTLLTUTTUWFMNY

d. ITIOIIUIALBCTIOTANTSISYDTOEOHRMLESENYROZTRNHDOTENONYSIEE
DIELAYOSSDMAIRENRBHDLFDNPMGHEATVRHNYTVITFEGHIIRREMIRUOTN
HSUIFGREALHNTKISTSSIEERGUHAOEDAEONDUENEEYPEURARSOVTSHNTT
COEMRNOPEOACIGAENEMFATUAAHNIEIDSHTIPGDSGPHDUDUETNEEAEOER
EALEHEMMNNRSYRHFGDIOHRSOAATMRHNYORTWSRIOVEUNINSNLSNMSNOE
OGNRHVMTTSEMFARTLEOIABHTUREREHPTEPLSDSRURARSGSTNDBTLUCIS
NOTMNHEEFTALTHIRBNMRNYSIEMENETEDIOFSYOAFSSEHYGRGRISYDTNO
HECELRESST

madness's book on classical cryptography
unit 52: permutations
last modified 2020-07-09

Unit 52
Permutations

Suppose we have an ordered collection of objects, such as [⌂, ○, ☼, ▲, ■, ♦]. A *permutation* of this collection is a reordering of the objects. For example, we might want to reorder them to [▲, ⌂, ○, ♦, ☼, ■]. We have moved the 0^{th} (because we start counting from zero) to the 1^{st} position, the 1^{st} to the 2^{nd} position, the 2^{nd} to the 4^{th} position, etc. We will write this permutation as (1 2 4 0 5 3).

We need a way of combining two permutations. Suppose we have permutations P and Q. The *composition* of them $P \circ Q$ is the permutation that is equivalent to permuting the set of objects first with Q and then with P. For example, (1 2 0) moves the 0^{th} object to the 1^{st} position and the 1^{st} object to the 2^{nd} position and the 2^{nd} object to the 0^{th} position, and (2 1 0) reverses the order of the three objects. The composition of the two, (2 1 0) ◦ (1 2 0) takes the 0^{th} object to the 1^{st} position and then keeps it in the 1^{st} position. It takes the 1^{st} object and moves it to the 2^{nd} position and then to the 0^{th} position. It takes the 2^{nd} object and moves it to the 0^{th} position then to the 2^{nd} position. The overall rearrangement is the same as (1 0 2).

The *identity permutation* is the permutation that doesn't do anything. It is (0 1 2 3 ...). The composition of any permutation P with the identity leaves P unchanged.

Every permutation has an inverse. The composition of a permutation with its inverse is the identity permutation. In other words, the inverse permutation undoes what the permutation does. Finding the inverse of a permutation is just like finding the inverse key for the monoalphabetic substitution. For example, let's find the inverse of (2 0 4 1 3). First, write the permutation under the identity:

(0 1 2 3 4)
(2 0 4 1 3)

Then, reorder pairs (shown with the same color) so that the second row is the identity.

(1 3 0 4 2)
(0 1 2 3 4)

We read off the inverse from the top row. The inverse of (2 0 4 1 3) is therefore (1 3 0 4 2).

The advanced reader will notice that what we have been describing is a group. A *group* is a set G and a binary operation $\cdot$ such that G is closed under the operation, the operation is associative, G contains an identity element, and every element of G has an inverse in G under the operation. The permutation group is noncommutative/nonabelian, which means that if we compose two permutations it matters in which order they are done.

We state without proof that any permutation of n objects can be generated by a series of exchanges of two objects. This also means that any permutation of n objects turned into any other permutation of n objects by composing it with permutations that each exchange two objects. We will use this fact implicitly when we attack ciphers that are based on permutations. When we attacked the monoalphabetic substitution cipher with a hill-climbing technique, we used exchanges to move from one key alphabet to another. After all, a key alphabet is merely a permutation of the regular alphabet.

The number of all possible permutations of n objects is $n!$.

Python tips

The `itertools` module has a function `permutations()` that returns an object containing all permutations of its argument. To make a list of all permutations of three objects, for example, we would use

```
from itertools import permutations
list(permutations(range(3)))
```

Be warned, however, that Python grabs memory for storing them, so if you try to list all permutations of more than around ten objects, it could cause problems for your computer.

We recommend that you use lists/arrays for permutations, rather than tuples, since tuples are immutable. You might want to modify a permutation, which is not possible with tuples.

Reading and references

Wikipedia:

en.wikipedia.org/wiki/Permutation
en.wikipedia.org/wiki/Permutation_group

Programming tasks

1. Write a function to find the composition of two permutations.
2. Write a function to find the inverse of a permutation.

Exercises

1. Find the composition of (5 2 3 1 0 4 6) and (6 5 4 2 1 0 3). Take the composition so that (5 2 3 1 0 4 6) acts first.

2. Find the inverse of (7 1 3 0 5 2 8 4 6).

Unit 53
Permutation cipher

The *permutation cipher* (or *block transposition cipher*, or *complete-unit transposition cipher*) divides the plaintext into blocks and applies the same permutation to each block. If the last block is too short, it is usually padded with nulls to fill it out. A *null* is a character that carries no meaning and is used as a filler. Often, ‘X’ is used, but other choices are underscore (‘_’) or random letters.

Let’s just do an example. Suppose we want to encipher this short message with the permutation (4 2 5 1 3 0):

```
THIS MESSAGE IS ENCRYPTED WITH A TRANSPOSITION CIPHER
```

First, we should break it into blocks of six letters and pad it if necessary.

```
THISME SSAGEI SENCRY PTEDWI THATRA NSPOSI TIONCI PHERXX
```

Then we apply the permutation to each block to get the ciphertext:

```
ESHMTI IGSESA YCERSN IDTWPE ATHRTA IOSSNP INICTO XRHXPE
```

The key of a permutation cipher is the permutation itself. Its inverse permutation is the key needed to decipher a text. Often, the permutation is written as a keyword. There are two ways to use a keyword to convey a permutation: with or without repeated letters. The way it works is that numbers are assigned to each letter of the keyword in alphabetical order. For example:

```
K E Y W O R D
2 1 6 5 3 4 0
```

If we are dropping repeated letters:

```
R E P E A T E D
4 2 3   0 5   1
```

If we keep repeated letters, we number them from left to right. So in this example, we number the ‘A,’ then the ‘D,’ and then the three ‘E’s:

```
R E P E A T E D
6 2 5 3 0 7 4 1
```

Breaking a permutation cipher by hand is done by “anagramming,” i.e., by rearranging letters and looking for meaningful arrangements.

Reading and references

Helen Fouché Gaines, *Cryptanalysis: a study of ciphers and their solution*, New York: Dover, 1956; previously titled *Elementary Cryptanalysis* and published by American Photographic in 1939; archive.org/details/cryptanalysis00gain; pages 9 and 64-65.

Programming tasks

1. Write a function to generate a permutation from a keyword. Allow for both ways of assigning numbers to letters (with or without repeated letters).
2. Write a script to find possible keywords from a permutation. Allow for both ways of assigning numbers to letters.
3. Write a function to pad a text given a block size.
4. Write a function or script to encipher a plaintext with a permutation cipher.
5. Write a function or script to decipher a ciphertext with a permutation cipher. Your function that inverts a permutation could be useful.

Exercises

1. Find a possible keyword for the example encipherment above.
2. What is the permutation corresponding to the keyword `PERMUTATION` if we drop repeated letters? What if we do not drop repeated letters?
3. Encipher this text with the permutation (2 3 0 1 4).

   ```
   ROUND AND ROUND SHE GOES WHERE SHE STOPS NOBODY KNOWS
   ```

4. Encipher this text with the keyword `POKER`.

   ```
   SHUFFLING A DECK OF CARDS CAN ON ONE HAND GUARANTEE
   A FAIR GAME BUT ON THE OTHER HAND PROVIDE AN
   OPPORTUNITY TO CHEAT THE DEALER CAN MANIPULATE THE
   SHUFFLE TO POSITION FAVORABLE CARDS INTO HIS OWN
   HAND THIS MUST BE DONE CAREFULLY AND QUICKLY TO
   AVOID DETECTION BY THE OTHER PLAYERS
   ```

5. Decipher this text with the keyword `REPETITION`. We will not tell you whether repeated letters have been dropped from the keyword.

   ```
   LTRELAITNSIIAOHIETETLATCERRYNUOEIHQUNAGSFIEAETPREITAIN
   DILTREELTNRSDWIO
   ```

6. Break this ciphertext by hand. What keyword was used?

   ```
   MCEOWLETTHEOOLODWRSATNFRIIOTPSHINPOCWWREESEBLEILIOPRXL
   WFGENACHFEOTCOMNOMSEPRIHLESLAWWFSEAAMONMUCSNNEOOLYHATE
   OVHELANHEINTMCIOGNETOHMNFUSFYHLHEELTSETRETTHFEOTTBXUET
   CEERNVTGAEHNRFEOHMLETROHSETRET
   ```

Unit 54
Heap’s algorithm

There are several ways to generate all permutations of a set of objects, and *Heap’s algorithm* is one of them. It swaps two elements at a time in such a way as to run through all permutations without repeating any.

Suppose we have n objects. The objects live in an array $[A_i]$ where i runs from 0 to $n-1$. In the following algorithm, the array $[c_i]$ holds a collection of n counters.

1. output A (the unmodified array)
2. set all c_i equal to 0
3. set i to 0
4. while i is less than n
    a. if c_i is less than i
        i. if i is even
            - swap A_0 and A_i
        ii. if i is odd
            - swap A_{c_i} and A_i
        iii. output A
        iv. increment c_i
        v. set i to 0
    b. if c_i equals i
        i. set c_i to 0
        ii. increment i

Python tips

Functions can be passed to functions. Here is a simple example:

```
def function1(x):
      print(x)
def function2(n,f):
      f(n) # call function f with argument n
for i in range(5):
```

```
function2(i,function1)
```

The result is to count from 0 to 4.

Reading and references

Wikipedia, en.wikipedia.org/wiki/Heap's_algorithm

Programming tasks

1. Write a function that takes an array and a pointer to another function. For each permutation of the array, the other function should be called with the permuted array as an argument.

Exercises

1. Wrap a script around your function and add another function to print the array. Use it to print all permutations of four objects. Check by hand that they are all present. There should be 4! = 24 of them, and they should all be distinct. Now be happy I didn’t ask you to use an array of five objects.

Unit 55
Factoradic numbers and permutations

Factoradic numbers (*factorial numbers*) are numbers expressed in mixed-radix form where the radices (bases) increase by one for each digit as we move to the left. Why this is related to factorials will become clear. The radix for the rightmost digit is 1, so that the last digit of a factoradic number is always 0. The radix of the digit to its left is 2, so that that digit can be 0 or 1. The radix of the digit to its left is 3, so that digit can be 0, 1, or 2. This continues as far as we need to hold whatever number we have.

Let's look at an example of converting a decimal integer to a factoradic number. Suppose we have 12345_{10} (the 10 subscript lets you know beyond all doubt that this is an integer in base 10). What is this number modulo 1? Try it and see. The answer is 0. Hence the last digit is 0. So far, we have $12345_{10} = ... 0_1$. Here we have placed a 1 subscript to indicate that the last digit is in base 1. We subtract that 0 from 12345_{10} to get 12345_{10}. This may sound ridiculous, by we are establishing a pattern. Now, the next digit to the left will have base 2. Now 12345_{10} modulo 2 is 1, so that digit is 1. Our number is shaping up: $12345_{10} = ... 1_2 0_1$. Subtract that 1 from 12345_{10} to get 12344_{10} and divide by the base to get 6172_{10}. The next digit to the left has base 3, and 6172_{10} modulo 3 is 1, so that digit is a 1, and we have $12345_{10} = ... 1_3 1_2 0_1$. Subtract that 1 to get 6171_{10} and divide by the base to get 2057_{10}. The next digit has base 4, and 2057_{10} modulo 4 is again 1, so now we have $12345_{10} = ... 1_4 1_3 1_2 0_1$. Subtract that 1 and divide by 4 to get 514_{10}. The base of the next digit is 5, and 514_{10} modulo 5 is 4, so we now have $12345_{10} = ... 4_5 1_4 1_3 1_2 0_1$. Subtract that 4 and divide by that 5 to get 102_{10}. The next digit has base 6 and we get a 0, so $12345_{10} = ... 0_6 4_5 1_4 1_3 1_2 0_1$. For the next step we have 17_{10}. The base of the next digit is 7, and we get $12345_{10} = ... 3_7 0_6 4_5 1_4 1_3 1_2 0_1$. Since that leaves 14_{10}, when we divide by 7 we get 2. We can go no further, and the result is that $12345_{10} = 2_8 3_7 0_6 4_5 1_4 1_3 1_2 0_1$. Some might write this as $2:3:0:4:1:1:1:0_!$. There is another way to find the factoradic representation of an integer by working left to right; we save its discovery for the reader. To convert our example back to a decimal integer, we use factorials:

$$\begin{aligned} 2_8 3_7 0_6 4_5 1_4 1_3 1_2 0_1 &= 2 \times 7! + 3 \times 6! + 0 \times 5! + 4 \times 4! + 1 \times 3! + 1 \times 2! + 1 \times 1! + 0 \times 0! \\ &= 2 \times 5040 + 3 \times 720 + 0 \times 120 + 4 \times 24 + 1 \times 6 + 1 \times 2 + 1 \times 1 + 0 \times 1 \\ &= 10080 + 2160 + 0 + 96 + 6 + 2 + 1 \\ &= 12345_{10} \end{aligned}$$

Notice that

$$\begin{aligned} 1_2 0_1 &= 1! \\ 1_3 0_2 0_1 &= 2! \end{aligned}$$

$$1_4 0_3 0_2 0_1 = 3!$$
$$1_5 0_4 0_3 0_2 0_1 = 4!$$
$$1_6 0_5 0_4 0_3 0_2 0_1 = 5!$$

etc.

Factoradic numbers are related to permutations in a natural way. Suppose we have *n* objects. There are *n* ways to place the first object. The name of the box into which we put the first object is a digit (as we call it, even if it is bigger than 9) from 0 to $n-1$. There are $n-1$ ways to place the second object, and we can record that choice as a digit from 0 to $n-2$. By the time we get to the last object, there is only one box in which to place it, so there are 0 choices, and the last digit is 0.

The mapping from factoradic numbers to permutations is called a *Lehmer code*. When the integers 0 to $n!-1$ are converted to permutations of *n* objects, they come in *lexicographical order*, i.e., in order as if in a digital dictionary and the numbers in the permutations are their letters.

Python tips

The pop() function deletes the n^{th} item from a list. The remove() function deletes an item based on its value. For example, this sample code removes the letter ‘X’ from the list, because it is the third element (counting from zero). Then it removes ‘B.’

```
letters = ['A','B','C','X','Y','Z']
letters.pop(3)
letters.remove('B')
```

Reading and references

Wikipedia, en.wikipedia.org/wiki/Factorial_number_system
and en.wikipedia.org/wiki/Lehmer_code

medium.com/@aiswaryamathur/find-the-n-th-permutation-of-an-ordered-string-using-factorial-number-system-9c81e34ab0c8

stackoverflow.com/questions/1506078/fast-permutation-number-permutation-mapping-algorithms

2ality.com/2013/03/permutations.html

Programming tasks

1. Write a function to calculate the factorial of an nonnegative integer (maybe zero). Avoid using recursion.

2. Write a function that converts a nonnegative integer to a factoradic number. We suggest that you store factoradic numbers as arrays of integers.

3. Write a function that converts a factoradic number to an integer. Be careful that only one object can be placed in any one box.

4. Write a function that takes a factoradic number and a length *m* and returns a permutation of *m* objects. Be careful to add 0 digits to the left if the factoradic is too short. Be careful that each factoradic digit *x* represents the x^{th} empty box, which is not necessarily the x^{th} box overall.

5. Write a function that takes a permutation and returns a factoradic number.

6. Write a function that uses your other functions to take numbers *n* and *m* and returns the n^{th} permutation of *m* objects.

7. Write a function that uses your other functions to take a permutation and returns an integer *n* to indicate that the permutation is the n^{th} permutation.

Exercises

1. Find 1000! How many zeroes are at the end of it?

2. Find the 101^{st} permutation of seven objects.

3. Which permutation (zeroth, first, ...) is (4 9 2 1 6 8 6 7 0 3)?

madness's book on classical cryptography
unit 56: brute-force attack on the permutation cipher
last modified 2020-07-09

Unit 56
Brute-force attack on the permutation cipher

To perform a brute-force attack on the permutation cipher, we need to be able to try every possible permutation. At this point, you already have the tools that you need.

Programming tasks

1. Implement the attack. Use tetragram fitness.

Exercises

1. Use your attack to break this ciphertext from the 2006 British National Cipher Challenge:

```
rlboy itdvs tennc rmaid toafl ubhle cneda nmoam nrdie
ficeh shoif sjmea hstsy nsiap sedsv mseel eopyl tndda
meoeb eopyl hndti ieedm artre eanne cettc oceyt hruea
yeebr iqrue ndrae eidrd tecdt eaokt hetsh asinp ndmie
aetmh uirng yenrd orocu damnm rdaan moeet frbka etohr
tdmie nrear nneaa eodst toeuk edadn yrsot hetsh fsiop
retfh fcehn aelte lotut etohn rproe gftao atemn tirnt
ihoet dorrl sisph etohn rpiom entca tionf pceer nntgi
eoalp hsocn eeisn teang tshsa drier tegar encco tirnn
rwhoe todfs rfhie rlsot ahdet repap ocaen ibfra ssthi
raqdu tionn dmhee reirt neaan oaics iintd woonn tchhi
tfhae ufeeo mpreo taaty oshmi dnmte deenp ruyao toest
eirna nrvye aveer ymnad rciun incso brdae zhlae ndair
cfeef intgi eetrh rnoof ouyno onrya mofuy ifaay oslya
liulw esawn hortt toenc arrya rotuy lrpie odafn oords
hgitn asihs yblel rroou irdge nnvoe dabro ietvh yocrt
rstio trhao lnieo onsco oomdm etrhe unvga ydabr amcmo
tonfd mahde lailr tdosr envci xxnxt
```

Unit 57
Hill-climbing attack on the permutation cipher

When we developed the hill-climbing attack on the monoalphabetic substitution cipher, we took a parent key, which was a permutation of the alphabet, and swapped two letters to form a child key. If the child key resulted in a better fitness for the plaintext, then the parent was replaced with the child. We will do something similar here.

We will make two important modifications to the algorithm. First, there are two ways in which we will want to generate a child key from a parent. One is by swapping two randomly selected elements in the key. The other is to roll the key a random number of steps. By this we mean to cut the key somewhere between two elements, and then to swap the front and back pieces. So, for example, if we cut (0 1 2 3 4 5) between the 1 and the 2, we would get (2 3 4 5 0 1); we have rolled the permutation two steps to the left.

The second modification to the algorithm is to add a margin of error. With permutation ciphers, it is easy to get trapped in a local maximum fitness. But we seek the global maximum. So we add a little leeway to the hill-climbing and allow it to take small steps toward lower fitness once in a while. A margin that works well for us is 0.15, and the probability of stepping downwards is taken to be 5%.

The algorithm does not find the length of the permutation. Usually, the key length divides the length of the ciphertext evenly, but not always. The key length must be input to the algorithm.

The algorithm:

1. set the best fitness to the fitness of the unaltered ciphertext
2. set an initial random parent key
3. set counter to 0
4. while counter is less than about 100 times the key length
    a. copy the parent key to a child key
    b. flip a coin
    c. if heads
        i. swap two randomly selected elements of the child key
    d. if tails
        i. roll the child key a randomly chosen number of steps
    e. find the plaintext by deciphering the ciphertext with the child key

f. calculate the new fitness of the plaintext
g. if either (new fitness exceeds best fitness) or ((new fitness exceeds best fitness minus the margin) and (we roll a 1 on a 20-sided die))
    i. replace the parent key with the child
    ii. replace the best fitness with the new fitness
    iii. set counter to 0
h. increment the counter

5. output the parent key

Python tips

The `shuffle()` function from the `random` module does what it says. For example, this block of code will fill an array with the integers 0, ..., 9 in random order:

```
from random import shuffle
x = list(range(10))
shuffle(x)
```

Programming tasks

1. Implement the attack. Use tetragram fitness. Experiment with changing the margin and the probability of stepping downward with some ciphertexts of your own choosing or making.

Exercises

1. Use your new attack to break the exercise from the previous unit.

2. Break this ciphertext.

```
STIOTASDGUTNINEFILMISTGEMEINNSATDESETASKSILUTTOLBSNLCI
TESYLNOELFVOETORMHRLYUCGIOVNEROOEKGTTPNETECOLERMOIRBDE
OMERGEITNTHERIPMWAIIDNRNKHEGMTOSESOWMNTNEHBETHCELSAKNO
DSHWULMNIDTEAEITDHVOUBWEOLDLNCGALITOLTESDTEHWEIMPAAIRA
GTUNSISJJPTTAUMHEOFTELNHTEADTSPNTATEHIOGTERWHHYTITRNOD
UHANUSROYOPOHUISYIYBORNGKEUSRNETHITNIGTSBTUITPVHIEELHS
CTTRUAETAHTRYILVLDRYIENSOUNESTAELOESTDTHERIPMWAAXAXGIN
```

Unit 58
Matrix transposition

A *matrix transposition* is the simplest form of columnar transposition cipher; hence, it is also called the *simple columnar transposition*. Encipherment is done by writing the plaintext into the rows of a matrix, then reading the ciphertext from the columns. Typically, the plaintext fills the matrix; if not, then nulls may be added. The key is the size of the matrix. Since we know the length of the text, we often only need one number to specify the key.

This cipher is also called a *scytale cipher*. The *scytale* was a rod around which a ribbon was wrapped. The message was written on the ribbon. When the ribbon was unwrapped, the letters on it would be in the same order as after a matrix transposition.

For example, let’s encipher this short message with a 6×8 matrix:

```
THIS MESSAGE WAS ENCRYPTED WITH A TRANSPOSITION CIPHER
```

The message contains 47 letters, so we will add one null to the end. Written in the matrix, it looks like this:

```
T  H  I  S  M  E  S  S
A  G  E  W  A  S  E  N
C  R  Y  P  T  E  D  W
I  T  H  A  T  R  A  N
S  P  O  S  I  T  I  O
N  C  I  P  H  E  R  X
```

We read off the columns to get the ciphertext:

```
TACISN HGRTPC IEYHOI SWPASP MATTIH ESERTE SEDAIR SNWNOX
```

Reading and references

Wikipedia, en.wikipedia.org/wiki/Scytale

David Kahn, *The Codebreakers: The Story of Secret Writing*, New York: Simon & Schuster, 1967, revised and updated 1996, page 82.

Programming tasks

1. Write a function or script to encipher a plaintext with a matrix transposition for a given size of the matrix. Allow for the possibilities that nulls are or are not added to fill the matrix.

2. Write a function or script to decipher a ciphertext with a matrix transposition for a given size of the matrix. Allow for the possibilities that nulls are or are not added to fill the matrix.

3. Write a function to find all factors of an integer. We are using small numbers, so it is not a problem to do an exhaustive search over all numbers less or equal to the square root of the integer.

4. Write a function or script to break a ciphertext that was encrypted with a matrix transposition. It would be reasonable to check all factors of the length of the ciphertext first, and then to check all other possible matrix sizes, in case nulls were not added. Be careful that you place the empty parts of the matrix correctly.

Exercises

1. Encipher this text with a 12×15 matrix:

   > Call me Ishmael. Some years ago, never mind how long precisely, having little or no money in my purse, and nothing particular to interest me on shore, I thought I would sail around a little and see the watery parts of the world.
   >
   > (from *Moby Dick* by Herman Melville)

2. Decipher this ciphertext with a 13×18 matrix. Is the matrix filled completely?

   ```
   TDFHEGATRAGOUOEWAYLDLIRHLTMWHNEAEETDLLAAAIDDDEDTFIOBPL
   TLTNPDHEFWUPKEEHEMOINEAREWWDESEWRCTNDAIABFSLNTEODERTSR
   ELAUYNHENEHHUNENPYIIXDAASCFCOANMIOBNHETHNRESSNUDHAHOHD
   FETTCAENILISEEEHKLSDMYSOEMNEEOUAASSFTECSTNRLNEPBHDEIOG
   VLDTIOIHB
   ```

3. Break this ciphertext:

   ```
   OONEOLDOSHEHMOSNKDDNISMTADIPFHEEHIHFEEIRRSAHEMFINITEDF
   DEMRILORMTSEHAFLAAEMOROSOADINSYDNDWONMEARHSDEAYYWAKITL
   HMIBDCBTLIVENRFOOSRITLOETEDGOTRUHOVIESGHDWWURRREWIOTLS
   TAHHBAILANDNOIPHBAELNBIDBESCDIEOTNESLKAEDTOETSUSGDFEEL
   LBHVOIITHRDOVBILYEEFFZHAERREATYABRFUEEPGEMRCTSREIALOLP
   OAEMKLLCDTNLFPERMDIAEIHDAYYTLNSSINNHGEINMTHEEAHNHDEHSN
   ```

```
DOHESDMEHEICTIGSMIRSTSFILFOLNWEENELOAOSAHUYTAENCSYMWUB
YELDOSMTOTAE
```

Challenge

Think about triangles.

```
IAAUSNPOETOHAUNRGSXTGNTQDSRDERENLSILXROGTUOOEAAYTDAOAEIOLHA
FNWNSARTTFNEDEARTEEDHBIRITDSSTEHOLCOOAIOHTBIFIFLARULBAOUAON
YRNTTSRRTMRGOOWSTDYIJTPUURTTATUHEINIOBTSARSDTOUTTHMHURTKAIT
NOINTSOKROOIF
```

madness's book on classical cryptography
unit 59: twisted scytale
last modified 2020-08-07

Unit 59
Twisted scytale

The *twisted-scytale cipher* was introduced in the 2011 British National Cipher Challenge. It is a modification of the matrix transposition (simple columnar transposition). Here's how it works. First, write the message into a matrix, with padding if necessary:

```
0    T  H  I  S  M  E  S  S
1    A  G  E  W  A  S  E  N
2    C  R  Y  P  T  E  D  W
3    I  T  H  A  T  R  A  N
4    S  P  O  S  I  T  I  O
5    N  C  I  P  H  E  R  X
```

Then, we roll each row to the left a number of steps equal to its row number (starting from zero):

```
0    T  H  I  S  M  E  S  S
1    G  E  W  A  S  E  N  A
2    Y  P  T  E  D  W  C  R
3    A  T  R  A  N  I  T  H
4    I  T  I  O  S  P  O  S
5    E  R  X  N  C  I  P  H
```

Read off the ciphertext by columns:

```
TGYAIE HEPTTR IWTRIX SAEAON MSDNSC EEWIPI SNCTOP SARHSH
```

The key for such a cipher is the number of columns.

We can modify the cipher to "twist" the matrix by an arbitrary number.

Programming tasks

1. Write a function or script to encipher a plaintext with the twisted scytale. Allow for an option to change the twist.

2. Write a function or script to encipher a plaintext with the twisted scytale. Allow for an option to change the twist.

3. Implement a brute-force attack on the twisted scytale. Remember to allow for different twists.

Exercises

1. Encipher this text with key 6 and twist 1.

   > One morning the old Water-rat put his head out of his hole. He had bright beady eyes and stiff grey whiskers and his tail was like a long bit of black india-rubber. The little ducks were swimming about in the pond, looking just like a lot of yellow canaries, and their mother, who was pure white with real red legs, was trying to teach them how to stand on their heads in the water.
   >
   > (from *The Happy Prince* by Oscar Wilde)

2. Decipher this ciphertext with key 7 and twist 2.

   ```
   OHVYLLINEYKBHEBHVMSUAMLRAFNATLNSABDRTPKONEETILEDAPSEED
   EEEITLDEYIGLTFELDATYEWHPDHERRHTOHSWTIFHDHWWTBRETINAYHR
   ROHDTHRAEETENELETWIHATXOASIAIHEEDHNHSIEIAWOBRESIHDORIF
   ECLSSAYOWRBTNSNTATHETEHNDVBMTEATLSASTXGTWIEWFDOEEEUADO
   LHUIERASEGOEIYHECEETHTAXHTOTAARGNGESTYFROEOFHERPWSWRGE
   ANODNTAOLT
   ```

3. This is the ciphertext from the 2011 British National Cipher Challenge. Decrypt it.

   ```
   TsormynhemuitlThesynpddmrdtcryefclseanpinptbsslenednul
   fteomatesdepaeestvsnysdsnmesirutnitgapsarmdxcdusrVerlr
   nitestixpurhepobakmsiarieiyrettFGcpgalitdwideddihdeivp
   otreesrtCFDdthaahthVaptetftddthrthapngteycitfehplceCra
   ebinatektadrytioosorngoxihIecfgestnoeeIesphaaetoreeiec
   aaabetvdrotliusiiWcrgmranhrnmuitiinongrvheenitiecrsicm
   IhcohenhotiiTtrystdXcuyatehyfhtboseatoneietictysievenh
   eraoomunnasctXsbyfafspelaeSxtiagsGwenepnefaanredsfwasl
   eitenhotocoipscWiseemoncblsstetiniccsevrnstldtfwmomuoi
   sTecinmwcTtirvrrheerneserrntchemofhooningiteoevecenaln
   owncgsdraieddanesanysnalntrnasoymennhispiaADisssIltwor
   hylsqnrsstnnhcaiTgsaeptelssrndsotndahrenairlaelinyoahc
   amhfouodhdXatnbesioeryaeseysoybhtuftheoitsdtpesyeetees
   sbgloianemachduimaertsrnonmocsyhaecoaetatimodeinsaseae
   forieitontenhavntieiatcinorltttemcyevcgmgudpamufthypnt
   garwtgetetnndFpuofclhorbesybudaitegfemlohuicdrineAtrei
   ithealVeptmepatcyednfmemlealaltisVDctdXrenfdesltocotgr
   uihrihelurhiehesphenomsecadsmmlsiatreerierisstdditpmat
   sinmpncimoeruusnntasIluyhuetexafeelciserslontiocsnkuha
   llsehaiustoietgesoreetderaeheleenegaaGhalaawekdosixiat
   ```

nsiptroudnedshbesgtDhesncessreanreorsnrtaptooplufnrptd
egrFTthecenloinaittufegatnesterfostmviTtiavaditarneane
linfiowvensx

madness's book on classical cryptography
unit 60: columnar transposition cipher
last modified 2022-01-13

Unit 60
Columnar transposition cipher

The *columnar transposition cipher* generalizes the matrix-transposition cipher by permuting the columns before reading off the ciphertext. The key is the permutation, which may be represented by a keyword. Repeated letters in the keyword may or may not be dropped.

Let's do a quick example. Suppose we want to encipher the following text with the keyword `KEYWORD`.

```
THIS MESSAGE WAS ENCRYPTED WITH A TRANSPOSITION CIPHER
```

First, we arrange the message into rows with the same length as the keyword:

```
KEYWORD
-------
THISMES
SAGEWAS
ENCRYPT
EDWITHA
TRANSPO
SITIONC
IPHER
```

The columns are then permuted. The permutation corresponding to the keyword is (2 1 6 5 3 4 0).

```
DEKORWY
-------
SHTMESI
SASWAEG
TNEYPRC
ADETHIW
ORTSPNA
CISONIT
 PIR EH
```

Notice that some columns are shorter than others. The ciphertext is then read off in columns.

```
SSTAOC HANDRIP TSEETSI MWYTSOR EAPHPN SERINIE IGCWATH
```

Of course, we remove the spaces to hide the column lengths from an eavesdropper.

```
SSTAOCHANDRIPTSEETSIMWYTSOREAPHPNSERINIEIGCWATH
```

Reading and references

Helen Fouché Gaines, *Cryptanalysis: a study of ciphers and their solution*, New York: Dover, 1956; previously titled *Elementary Cryptanalysis* and published by American Photographic in 1939; archive.org/details/cryptanalysis00gain; chapter VI.

Practical Cryptography, practicalcryptography.com/ciphers/columnar-transposition-cipher

Abraham Sinkov, *Elementary Cryptanalysis: A Mathematical Approach*, 2nd edition, revised by Todd Feil, published by Mathematical Association of America, 2009; www.jstor.org/stable/10.4169/j.ctt19b9krf; chapter 5.

William F. Friedman, Military Cryptanalysis, Part IV: Transposition and Fractionating Systems, Washington D.C.: U.S. Government Printing Office.

American Cryptogram Association, www.cryptogram.org/downloads/aca.info/ciphers/CompleteColTransposition.pdf and www.cryptogram.org/downloads/aca.info/ciphers/IncompleteColTransposition.pdf

Fred B. Wrixon, *Codes, Ciphers & Other Cryptic & Clandestine Communication*, New York: Black Dog & Leventhal, 1998, pages 152-153.

Fletcher Pratt, *Secret and Urgent: The Story of Codes and Ciphers*, New York: Bobbs-Merrill, 1939, chapter V, sections III-IV.

Programming tasks

1. Write a function or script to encipher a plaintext with a columnar transposition cipher.
2. Write a function or script to decipher a ciphertext with a columnar transposition cipher. Be careful of where to place empty spots in the matrix, if any.
3. Make a copy of your brute-force attack on the permutation cipher and adapt it for the columnar transposition cipher.
4. Make a copy of your hill-climbing attack on the permutation cipher and adapt it for the columnar transposition cipher.

Exercises

1. Encipher this text with the keyword `PERMUTE`.

> “You are old, Father William,” the young man said,
> “And your hair has become very white;
> And yet you incessantly stand on your head—
> Do you think, at your age, it is right?”
> “In my youth,” Father William replied to his son,
> “I feared it might injure the brain;
> But, now that I’m perfectly sure I have none,
> Why, I do it again and again.”

(from *Alice’s Adventures in Wonderland* by Lewis Carroll)

2. Decipher this ciphertext with the keyword `COLUMNS`.

```
WNIONOIENNANNACTTEITTAITWLWCAHOEEHANSOYKSGCLSKTHLCSYEO
EAEGEAAEEOBNMOAEBEWGATLPEISEFEUMANDBOSAWUPODTCREAONLDR
TTDDAETTNTIRQSTAYDDENOCELIUORAOTCWOIWOTSALIMSEANCADCNT
ASGLMIHHRNHYVNLHLIAYTSNMOGETNODHOIUCIISIECSFTWLNRFUSMA
KSRGHKLSJINOKECWAPODBEWEUICUFTPUCRSDFSLDEFNBANEEGHI
```

3. Brute-force this ciphertext.

```
OTENAYOSTAEPFLEUSEHKTFQEEIROGOSAEUCODWNREETFUTRYFOTGHT
AYETNANTELETCBMTMSEAAMINFHUSRCEAAEASLVALSNNASNUEUSTEBS
KPTLBEVAHHUTUOEUFHOEELNINSEETELUEEOIRLWEOHTRHHMSNYURAE
KYYTSPSOELOUDTRSISZRTEPETEEHOESITAEEHSNBSOENYEATSETVTO
MASHNMFEDONUTHFEEAIGISOEAUNMEMHEOOPOSEIDEEOEOEHHLNAADO
SMIHTTQEMORJETORSEOSEEUGIHNJNVMTEUTMYGEUDTRSIUYYDLOUEI
SLOLYCRPDENHYUYYGTOITHNCROVTIOSZYGITBEGIUHOIHROROTIDOA
YNHOEOSIESNREDG
```

4. Break this ciphertext with your hill-climbing attack.

```
UHOEIHSHIBMEWTRNWBETDHTOHSYSHTHNNBETWEROHTWAOTDDROTLCO
WHOWYETEIFWACTENEDEEIEQTSTRCDFMIESTLRSELITUIRTDRRETRLD
CAAETOBECDMATAOTWNTSGLIDVSKISAGTSCTEENHWNLESUEDBSTFEST
RHSESWLEWYYLEUULSSIRNHEEDYLDRRACTHTNGSAENIOEWEIHMEOAOP
HUTEDCDTHENARHUEAAKSLGICNWRFGAANINANAMESEABSHIDAEIHLET
NIUBEOHAOEEAHAIYFEHASUHSRYUSOACAEBEIRHRRLANAEWGATPANEU
IAHELASATNWFFDUTRDHLTIITPADETRBHSLAANRAAOONOHEESNEIIHH
DRTTLODTIDUAIFGMSNIAEHUOSTEEAOVDMIEETBARSUNABEUNYDFOAE
OTHUHEEKOAYITOCTOIWLRYESEPTHEOOWSACEEUACTSIEWDELTPNAHY
WOTTUVDHSSGSNTITHYOHOOBASDSSDTHOSGLUTTNTIORRYNEESDATCE
NEDCOCENDTNSLVDWBOESENRISNWKHSHIFFEYEINWVSIAAUSAATOTAB
IAEHTAAIEEWLTLEBETHHRETHAIHHWLGIETELMNHHOAWMOIEAIKCHGS
```

GUSHTDDSUIASWEDSEAODEDSOIKREGERNSTRTPWLLHELYTUNSDSNAAV
DSOPRYYPESHYGLOSENDBTESSPNAAKTNHNIEOINC

madness's book on classical cryptography
unit 61: double columnar transposition cipher
last modified 2020-10-14

Unit 61
Double columnar transposition cipher

The *double columnar transposition cipher* is the application of two columnar transpositions in sequence. Its key is two permutation, which can be represented by keywords. It was a popular cipher during the two world wars.

A hill-climbing attack on the double columnar transposition can be constructed by modifying our attack on the single transposition. There will be two parent keys (one for each transposition) and two child keys. In each step in the key space, we randomly choose one of four options: swap two elements of the first child key, roll the first child key a random number of positions, swap two elements of the second child key, or roll the second key a random number of positions. Unfortunately, there is no way to determine in advance what the two key lengths are, so we often have to try several before we succeed.

Reading and references

NOVA Online, www.pbs.org/wgbh/nova/decoding/doubtrans.html

Wikipedia, en.wikipedia.org/wiki/Transposition_cipher#Double_transposition

Solomon Kullback, General Solution for the Double Transposition Cipher, Washington D.C.: U.S. Government Printing Office, 1934, www.nsa.gov/Portals/70/documents/news-features/declassified-documents/friedman-documents/publications/FOLDER_439/41751169079035.pdf

Fred B. Wrixon, *Codes, Ciphers & Other Cryptic & Clandestine Communication*, New York: Black Dog & Leventhal, 1998, pages 157-159.

David Kahn, *The Codebreakers: The Story of Secret Writing*, New York: Simon & Schuster, 1967, revised and updated 1996, pages 301-303.

Programming tasks

1. Write a function to encipher a plaintext when given the key.

2. Write a function to decipher a ciphertext when given the key. Keep in mind that the two transpositions are done in opposite order when deciphering.

3. Implement the attack described above. Copy the attack that you made for the single transposition and make the appropriate modifications. Experiment with the margin; can you manage without it?

Exercises

1. Encipher this text with keywords `SAMPLE` and `KEYWORD` (in that order).

   > At the Carlton news stand West bought two morning papers, the Times for study and the Mail for entertainment and then passed on into the restaurant. His waiter, a tall soldierly Prussian, more blond than West himself, saw him coming and, with a nod and a mechanical German smile, set out for the plate of strawberries which he knew would be the first thing desired by the American. West seated himself at his usual table and, spreading out the Daily Mail, sought his favorite column.
   >
   > (from *The Agony Column* by Earl Derr Biggers)

2. Decipher this text with keywords `TOMATO` and `SOUP`.

   ```
   EOONPGAEPOGOCPEPCRWEUNNELMRDNIOOLFEIAOTTASCRCCATERWDGO
   SNNSKNGIDOEIDFEELLPALIHNNTISYDNNRSHPFEANOINDOEIELVDERS
   MLOIRRDROCSLYETEEWFRGEIEHWILIERNALNOSHCIHACEEBIUEBROPV
   NNREDEOPRMPIROOEAMHUIOCAESIOJSSSOODNROTKONAESONAADWRIG
   DOLPEERETOEERTAPHETADEOYRTFPKOLGDDHCTTDETVAAELLSCUAEDT
   RRRUPAALSUOECDLTCDNPNCQNDARSAIUANEPSEOOSAKYYBIRRORRLHK
   IFTDCEDNPGOCIRETROYTCOOOTYOGNODHTLOSUUGPPNBACONWSTBRHG
   MPEETEDENIMSCGTAETAOOUAUNPDTEDEOR
   ```

3. Break this ciphertext. The key lengths are five and six.

   ```
   VNRWGNCARTTNONRRWOODRLIOOSWPAEHTHUNDCTTIAHIRLASDIHNTHI
   AHOHANMSNFTEUECSNDEKRROSFGNUEFSRFODGIHONNIEETHEROAOALB
   REEUIGONFESINAMTOFLBETTWWNSNAFMTOECLEEAEAEHTYTNIGVETAR
   RNILTOEAHENATHGLIDDJDADGHWTOTRSEOATTSLDHNHFWCOLEIWIEDI
   DAAARSOBRATIHGFSEDAROCORAISYRLVTIEFTSAERERFNLRATVTSNTR
   AIWNSESENVQTDODTSOOLLNGFAFIMBAARENEOMEIEWEUKUONLOSISAL
   HOVINSTBEESLRETHBDISESGDAEVTORGHRAEDOOIGRHDSAOITITOEIA
   THTATBHUALTAISHBTLTHDERAETTJKEDOELCEWETOSREYEEDGTGTSRW
   THSLIWNWNADNOEOPOETOIIEONOAWCYNITITEOTBWEOAFRRIHEAFAHP
   ETOIRTTIOSADAAOHOAHLIWETYNDEESIHICNCOOEFUAEATWFAETTKEH
   ESOOOITUINTSIAJLDEHWUWEYS
   ```

4. Break this ciphertext.

TTFNOEINTDDRNTBEOSROHOYEIAEFAIEYFEOREECTTHSITEROTOPPTN
RPROCGHSNIFTJPOEEGHSCPUKENDDIHSIIGETILSEOETDIEOPHIHBIA
NDDENMUITMHNTEOLTCEAACSPTOUIETBHTOUAOOASOOHEUVRRHRNOTN
YRNMCIOOAEOGHEEMTTOSNDTENDIAORNSVOAESDNROTPLCHTTMRAHHO
SCNIMCFEMRLMNGIERETTEDTEMZBTMRHPNTVHESGSHEEEIVUEHEAYUI
PLHIEAITHRLHNNOWNEEEWTEDEPRTTRUBIIETOONOAAEITSEFSODDLM
CALWOERTHUTIYELDTOBWTTVDIATURTRRYIOYIOTEIFTPSTORTAWFOG
IEOSTESDOSBAIIIO

Unit 62
Nihilist transposition cipher

In the *Nihilist transposition cipher*, the plaintext is written into a square matrix, with padding if necessary. Rows and columns are both reordered with the same permutation, which can be represented by a keyword. It does not matter in which order the two operations are performed. The ciphertext is read off either by row or by column. If the plaintext is too long to fit into a grid whose dimensions are the length of the key, then it is broken up into blocks, each of which is enciphered separately.

Do not be confused and assume that this is a permutation cipher followed by a columnar transposition. Both the permutation cipher and columnar transposition shuffle the columns when the text is laid into a matrix. However, in the Nihilist transposition, rows and columns are both reordered.

Reading and references

Helen Fouché Gaines, *Cryptanalysis: a study of ciphers and their solution*, New York: Dover, 1956; previously titled *Elementary Cryptanalysis* and published by American Photographic in 1939; archive.org/details/cryptanalysis00gain; chapter IV.

American Cryptogram Association, www.cryptogram.org/downloads/aca.info/ciphers/NihilistTransposition.pdf

Fletcher Pratt, *Secret and Urgent: The Story of Codes and Ciphers*, New York: Bobbs-Merrill, 1939, chapter V, section VII.

Programming tasks

1. Implement an encryptor for the Nihilist transposition. Allow for both choices of how the ciphertext is read from the matrix.
2. Implement a decryptor for the Nihilist transposition. Allow for both choices of how the ciphertext is read from the matrix.
3. Implement a dictionary attack on the cipher.

4. Implement a hill-climbing attack on the cipher. Use parts of your attacks for the permutation cipher and columnar transposition, and make the appropriate changes.

5. Now suppose the matrix is no longer square, and that rows and columns are reordered with different permutations. Copy your hill-climbing attack on the double columnar transposition cipher and modify it for this new cipher. Again, allow for both choices of how the ciphertext is read from the matrix. Also remember that a long plaintext may have been enciphered in several blocks.

Exercises

1. Encipher this text with the Nihilist transposition with the keyword `BOYNIHILIST` (do not drop repeated letters). Use 'X' for padding. Read the ciphertext off by rows. You will need to encipher it in two blocks.

   > But outside of his being an enthusiast and a lover of liberty, he was not known, and had never taken any prominent part in any of the social or political movements of the day, beyond sympathizing with the struggles of the working men and women of the world in their struggles to better themselves.
   >
   > (from *The Boy Nihilist* by Allan Arnold)

2. Decipher this ciphertext with the Nihilist transposition with the keyword `SIBERIANGULAG` (do not drop repeated letters). The ciphertext was read off by columns.

   ```
   RHFHLADAEXGTIHDCOFNCRDDMETSHHPHOIANXEDMOOERECSYDXRFEUE
   DTUHAHTGNAHEHANFCOOEDANFIIEOLDHYEEEATOVEICWRGONIFRFUEI
   DHMTIPORBNARMTTELDXMHMJAPWIEMUVIKDOESRMTNEWAYVNOTLSSIA
   YHSAFIU
   ```

3. Break this ciphertext with your dictionary attack.

   ```
   OIUDYOHENTTIAVAFMELLWIORILDWANDOOULYYOSTRUONEYUBRTBELI
   ITANYCEOUSCAANUMFHLIHETTNDSAVEUKYOENHAWTNOUBDODAWHNDTA
   NDNAMENDMEWOSOEDDRVERAFBRTBELIUNFHYOLLWITHWEUSYOSTRURT
   AIDFANDUENEPOUNYPOSBERETNOGKURODYBERTPNSYIBETOAREVUETR
   NEEALBATYMASANIXOWILTWDIINPRWSSTMACEMSHICHGTINEENDDAIN
   OAPRAPORRFTEOFUTYOHTIGLRYOLLWISSPAURALRTPOOUTYHAYOVEHA
   EIOPELHAISNHETACPLNVSEHIARWEUSTOMETONDHAORXXITLELSEEON
   EEOSETLOALNDNONGWI
   ```

4. Break this ciphertext with your hill-climbing attack. What are possibilities for the keyword?

   ```
   THTATEMMORAUBTILILKEOTYODVETRENUTRNOAHNTIHIILATSIHNNIS
   ATNOISILOUTAASNDSSSASXSWXXSOHSETADLLFROTOOEFLL
   ```

5. Break this ciphertext which was encrypted with the cipher described in Programming Task 5. What are possibilities for the keywords?

```
AMSAIBSOIHERQKOCNEISTATHTERDUSERFPAPMTETRWIOTERSIUATTS
UIEMCAITCUBUEPVIHWANONOEETCNEOCONTNASTHISTSHSEEBAYENIE
LTNULNUDLRUOFYOGOAMNOSIHNTYOPLANOGFNIIRTWREVRNITNMERSD
AOONADSGFFEIDNAYMYNAYMOASPYAAFCERNISDNTNWERFFEIDDNEETO
TTECERAEHISNIEDOURTIONTDRIIAETKHOOOFLTATRHEDEHBTHIHCWN
BTESUIARTAATCTASWMOLWTHUBEKNATMMOOCRLEACSNTOCISNEHROIT
SHOITNTSONOIREAGSOTTSTCIURVTACATSRBAEJSOTSSIERFOANOEIS
ROMIPEHOTTSTTDONAAHDTISDCAEROMTSECWNAONYNAHTETSCUTDOWN
HOITNIEITNRIFTROOSLKIOOLSIISHTAEKAHBCIEEDNSHTIEKAMNAHY
AWSOHSOOHIHCWTDNTIFHGTIMOTDEVESAUOONFOTHDALUATHARVIELE
EMSOSSKAOORRAAWLASOCDNBAHNODCELAWRETSSACADGNAESVTNROEF
RHTEAETNIAUTOKNOHWOONTAQERITDSUFODYTNAWMALEFSMAESLPCUH
SWHEIKADLMUOHEOHTAPRERROTFETTONAROPDATSEEMITOPEAODLAXX
XXXERCEHEXDIMSFOON
```

Unit 63
Railfence cipher

In the *railfence cipher*, we write the plaintext down in a zig-zag pattern that runs over a number of rails. With no offset, we begin on the top rail. With an offset, we skip some positions before beginning to write the text. The ciphertext is read off one rail at a time.

Let’s do two examples. We begin with the plaintext:

```
THIS MESSAGE WAS ENCRYPTED WITH A TRANSPOSITION CIPHER
```

The first example will not use an offset. Let us take four rails. We write the text onto the rails:

```
T-----S-----A-----Y-----I-----A-----I-----I----
-H---E-S---W-S---R-P---W-T---R-N---S-T---C-P---
--I-M---A-E---E-C---T-D---H-T---S-O---I-N---H-R
---S-----G-----N-----E-----A-----P-----O-----E-
```

The ciphertext is read off in rows, starting from the top rail.

```
TSAYIAIIHESWSRPWTRNSTCPIMAEECTDHTSOINHRSGNEAPOE
```

Now let’s repeat the process with an offset of five. We skip five positions when writing down the plaintext.

```
•-----H-----S-----S-----P-----T-----N-----T-----P---
-•---T-I---S-A---A-E---Y-T---I-H---A-S---I-I---I-H--
--•-•---S-E---G-W---N-R---E-W---A-R---P-S---O-C---E-
---•-----M-----E-----C-----D-----T-----O-----N-----R
```

Read off the ciphertext to get

```
HSSPTNTPTISAAEYTIHASIIIHSEGWNREWARPSOCEMECDTONR
```

Reading and references

Helen Fouché Gaines, *Cryptanalysis: a study of ciphers and their solution*, New York: Dover, 1956; previously titled *Elementary Cryptanalysis* and published by American Photographic in 1939; archive.org/details/cryptanalysis00gain; page 12.

Practical Cryptography, practicalcryptography.com/ciphers/rail-fence-cipher

Wikipedia, en.wikipedia.org/wiki/Rail_fence_cipher

American Cryptogram Association, www.cryptogram.org/downloads/aca.info/ciphers/Railfence.pdf

Fred B. Wrixon, *Codes, Ciphers & Other Cryptic & Clandestine Communication*, New York: Black Dog & Leventhal, 1998, pages 139-141.

Programming tasks

1. Write a function or script to encipher a plaintext with the railfence cipher and a given key consisting of the number of rails and an optional offset.

2. Write a function or script to decipher a ciphertext with the railfence cipher and a given key consisting of the number of rails and an optional offset.

3. Implement a brute-force attack on the railfence cipher. Note that the number of rails needs to run from one to the length of the text, and the offset needs to run from zero to twice the number of rails minus two (with a maximum of the length of the text). Use tetragram fitness to determine when you have found the correct plaintext.

Exercises

1. Encipher this text with four rails and no offset.

```
DO YOU SUPPOSE THERE IS ALSO A PICKET-FENCE CIPHER?
```

2. Encipher this text with five rails and offset seven.

```
HOW DO YOU SUPPOSE A PICKET-FENCE CIPHER WOULD WORK?
THE PICKETS ARE VERTICAL, AND THERE ARE A LOT OF THEM.
```

3. Decipher this ciphertext with three rails and offset two.

```
YEYITTIAAELSEEOIHAESEBERADUTOORALBLEEHTHSSHTRIFNEOKLKU
LSYUHNTELITXILKTEABDIERPEAONIDULEVAIWALCOINSTKPNTIHRWW
PRD
```

4. Break this ciphertext from the 2014 British National Cipher Challenge.

```
PRSAO EGERA UIADM WEHDN ISNRA SAWUA AESSR EFGDO SOGVO
RBEEE AARTE SCTDF MENUI BRTTL MEYTU MTMEU AIKWH UTKWE
RWAHM NPWRA EESON ONESE BATOI HACIN EETBR OTADA KTGFE
ESYIO FLTTL STIIA EOSVI EONSR RTAUP MNNOA ENCOC NUVRS
CLVDR GCTAI IHRIC IAIHR SDUOM RLEMC RNGLE OMARF HIUEW
HALCS ASRAC UFRAW WSMEH ULSTO AOHCE LETMT OILSE PDMUM
TPTRS LYRHH NTPAN WPMOA DPPDW BESEO ASSLT MLPES LETUN
CORER LCLIT AOSVS INIIF WSEAF ORTAA DUYEN ENONN SOPFH
ONTWK OERTC SLYVO EIOHL UFOEI OETST HTSBR ENEVE AOUEP
GIEES OBDUO RSFEE RCDYA DUTAE PEADR DIGSE EBFUO GGOPO
GALYF EWSOE EMDNT OHREB HAAES NEWOR GNFIA ULNLW ADUEO
DCOTR ARGVU ENEWH IERTL AUILM SONIO TMUIN EWAIU EWLOE
RSTTT ISDRS ASNUS SIESM ERDHE TRYRH PNLRT EREAD MREDE
BNNTR NENWM OUTRD OSANE OWOMC GIDCI ASAON TIIOI ASCES
ISSUP CRMOY BRINE YWEEL AYLEW TYRTI LHSTO
```

5. Break this ciphertext.

```
MICYRLHHTAHTLZHAENOHIEALEOBFGALETETOSAZGTAFHRAGATEFLVN
YATHTETARFRNDINIEDDNGSNCGPOTOEEPIGTFMLEWNIEOERGGTEESER
ILFTNBNEIKIASTSRVPZGPEOINIALPLGNOINWTTSWSMRAYIRSSZTNHD
AOEEIENRNWTTSODOAOSLCSRCTEFOESLYALIATRENISDHNKCFGOHUOR
ITLPEUHREAINEELTGEDORT
```

Unit 64 (optional)
Redefence cipher

The *redefence cipher* is a modification of the railfence cipher in which rows are read off in a different order. The key for the redefence is the inverse permutation of the rows (we use the inverse so that it lists the order in which they are read) and the offset with which the plaintext is written in.

Here is the example from the previous unit. We will take the key to be (1 2 0 3) and 5. The plaintext is written down with the offset:

```
0     •-----H-----S-----S-----P-----T-----N-----T-----P---
1     -•---T-I---S-A---A-E---Y-T---I-H---A-S---I-I---I-H--
2     --•-•---S-E---G-W---N-R---E-W---A-R---P-S---O-C---E-
3     ---•-----M-----E-----C-----D-----T-----O-----N-----R
```

The rows are read off in the order 1, 2, 0, 3 to get the ciphertext.

```
TISAAEYTIHASIIIHSEGWNREWARPSOCEHSSPTNTPMECDTONR
```

Reading and references

American Cryptogram Association, www.cryptogram.org/downloads/aca.info/ciphers/Redefence.pdf

Programming tasks

1. Write a function or script to encipher a plaintext with the redefence cipher and a given permutation and offset.
2. Write a function or script to decipher a ciphertext with the redefence cipher and a given permutation and offset.
3. Implement a brute-force attack on the redefence cipher.

Exercises

1. Encipher this text with the redefence cipher. Use offset 2 and permutation (3 2 0 1 4).

   > I have committed sins, of course; but I have not committed enough of them to entitle me to the punishment of reduction to the bread and water of ordinary literature during six years when I might have been living on the fat diet spread for the righteous in Professor Dowden's Life of Shelley, if I had been justly dealt with.
   >
   > (from *In Defense of Harriet Shelley* by Mark Twain [Samuel Clemens])

2. Decipher this text with the redefence cipher. Use offset 3 and permutation (2 1 0 3).

   ```
   HJCTIOIILRTRTUPAESFAFAIAYNPTOWODRTININWHASLCRLNNECSLYA
   EGRYNPETPUEHRWIUTVTYAPNNHEIFHOFITDYEBTFSOSOUTEUTRRPDEO
   WRMLTSAONSOWNCRNOTOHCYAYCEGDASCSUIVDYMNDOSNITOTUNCMIEA
   LLUUOEVNOANADATOOBTSOEENROITIHUECIIEONCUFNBATECFIHNBLS
   EGTRNTEHKLAUNRSRMRDTHEADSMIUABELDEARAREHSIFOTNTTSA
   ```

3. Break this ciphertext.

   ```
   AITEENSTIMUTOOSEEEIYDEAATMAATWTNTTSDEIRAOESPSOIIFATRIC
   UINHDSFATRAIFRRWITDAVLHYENLREANARGYLTHGEOPENDHYINUOQAI
   CIOINFHMSNTANLMFCOEACOLDEMPPLESWRHTUANPUEOOHCSYSNHTILT
   NJCISDUOSRGNIADOISUTHLVLYETSIEPINREEVOHIIIDHRFLFLYASOA
   FGDMIAOALAENOUIKNSOTELRTOOCLTTSEARSNRCINTSRIBNIELTONCI
   LTSTRWHOISOTTWBSTETSANTEBLHINNSPTDNAELEACUHNNVPTAOTSET
   EDOWOTA
   ```

Unit 65
AMSCO cipher

The *AMSCO cipher* was invented by A. M. Scott. How it got its name is still a mystery.

To encipher a text with the AMSCO cipher, we are going to write the text into a matrix. We must first decide whether the first box should contain one or two letters. After that, we fill in by rows in such a way that when looking across a row or down a column, entries in the matrix alternate between holding one and two letters (the box containing the last letter need not be padded to hold two letters). The message is not padded, so some columns may remain incomplete. For example,

TH	I	SM	E	SS	A
G	EW	A	SE	N	CR
YP	T	ED	W	IT	H
A	TR	A	NS	P	OS
IT	I	ON	C	IP	H
E	R				

The key is the order in which the columns are read off to form the ciphertext. If our key is (2 5 0 1 4 3), then the ciphertext for our example is

SMAEDAON ACRHOSH THGYPAITE IEWTTRIR SSNITPIP ESEWNSC

Notice that the key is the inverse of the permutation that acts on the columns. The key can also be represented by a keyword.

Reading and references

Helen Fouché Gaines, *Cryptanalysis: a study of ciphers and their solution*, New York: Dover, 1956; previously titled *Elementary Cryptanalysis* and published by American Photographic in 1939; archive.org/details/cryptanalysis00gain; page 51.

American Cryptogram Association, www.cryptogram.org/downloads/aca.info/ciphers/Amsco.pdf

Programming tasks

1. Write a function or script to encipher a plaintext with the AMSCO cipher. Allow for the choice of using one or two letters in the first place.

2. Write a function or script to decipher a ciphertext with the AMSCO cipher. Allow for the choice of using one or two letters in the first place. Be careful how you handle incomplete columns.

3. Implement a brute-force attack on the AMSCO cipher.

Exercises

1. Encipher this text with keyword `YACHT` and two letters in the first box.

   > This unlikely story begins on a sea that was a blue dream, as colorful as blue-silk stockings, and beneath a sky as blue as the irises of children's eyes. From the western half of the sky the sun was shying little golden disks at the sea—if you gazed intently enough you could see them skip from wave tip to wave tip until they joined a broad collar of golden coin that was collecting half a mile out and would eventually be a dazzling sunset.
   >
   > (from *Flappers and Philosophers* by F. Scott Fitzgerald)

2. Decipher this ciphertext with keyword `PARADISE` (drop repeated letters) and one letter in the first box.

   ```
   AMRAMEEOSSTPRORUNEELRYREDERRCANIHIGDAEEDETHIMOSORLDTME
   OGAVIETDNGNIVEYRVGMUINGREASFYOOSANTLELAWCHSINOYNNEWEFY
   OGOFE
   ```

3. Break this ciphertext. What might the keyword be?

   ```
   EHEBWIASHITLONWWEDNEROEYEESENDETERIMPEETHSSAETSTEGNTTA
   RYKZIHECSTHHSPIESUHEHHALTHDETESNAPPRCEEASMGAORORSTTNDN
   DRTRAFEPRYBPASNMAINSYPIHHOUERORNEFITAKUCOAESSERFITEFSC
   GHBREIMERICADSHHIGIPOGAHPBENSTIBSUCOLEINDAIPSJEPAGRFTF
   INRSTAGTIIMIWITTOROKGSNRTATOEILHPAEUTEDURARARINCHTELCF
   O
   ```

4. This ciphertext is from the 2015 British National Cipher Challenge. Break it. What is the keyword?

   ```
   UOCAD EDMMA EDAEC RRSOI COIPI IFEBO BDUSI SDSBE ENTTY
   ANETA DCOUD KOOTT RBELE DRAOA USYYO HISNL EDDSR ISHRS
   HEEOS ETROS TLSEU NRSCC OUWRI MOLHN STERS EDEEA RINTE
   WPENN OMSFT AITOF TOILL CAPEC ESEPN DUETN DEEUS ESOMF
   AKOGT INLGE LYARU ITSIC RIOIR SECOR ETNEU EATMT ALIHL
   EICRV EASME NCIED ATHEO LKWHE OHNWO NFIMH THUBE AIBND
   ```

ATTHU EKTDO OECTW ITABA OULCA QWIUP IYPOC ODEEO ONSHI
TCIEE ATAIA CLTTA RCHGH OFHET DPAEC HMAHV EHLOL YARLE
IKEDR ODTEI DYSKE EYCFS HFOFE HTGEJ NDBBR YPHON IOSRO
LMREF RWFII GHYPH NMEHR ELASO NDEIV TDTIR HEROT URECH
ORHEG YNDED MEPYT WVIID OMOUT YAION AUFUC ATWOU OKTEF
LTHMU TGNEC TITAR AOWEN NGSST EFRRE SCLFS PSATO EHIHE
LASTV CHIFI IUTET EYEEC STNHE OILDS TNEPM ROSTT SRAFL
LRHED ISESS RETIE NASU

Unit 66 (optional)
Myszkowsky cipher

The *Myszkowsky cipher* is a modification of the columnar transposition. It requires a keyword that has repeated letters; otherwise, it degenerates to a regular columnar transposition. The plaintext is written into a grid with the same number of columns as letter in the keyword. The columns are labeled by numbers representing the alphabetical order of the letters of the keyword, where identical letters have identical numbers. To read off the ciphertext, we start with the first row and read all letters in a column labeled 0. Then all letters in the second row in a column labeled 0. When we have finished all rows, we start again at the top with all columns labeled 1. Then 2, etc.

An example couldn't hurt. Let's encipher this short message with the keyword `TATTOO`.

`THIS MESSAGE WAS ENCRYPTED WITH A TRANSPOSITION CIPHER`

The keyword `TATTOO` is converted to numbers 2, 0, 2, 2, 1, 1, because 'A' is alphabetically first (zeroeth), 'O' is next, and 'T' is last. We write the plaintext into a grid with those column headings. There is no padding.

```
2  0  2  2  1  1
─────────────────
T  H  I  S  M  E
S  S  A  G  E  W
A  S  E  N  C  R
Y  P  T  E  D  W
I  T  H  A  T  R
A  N  S  P  O  S
I  T  I  O  N  C
I  P  H  E  R
```

We read off all of the letters in 0 columns: `HSSPTNTP`. Then all 1 columns: `ME EW CR DW TR OS NC R`. Pay close attention to the order of the letters. We take all 1 columns by rows. Now, all 2 columns: `TIS SAG AEN YTE IHA ASP IIO IHE`. The cipher text is

`HSSPTNTPMEEWCRDWTROSNCRTISSAGAENYTEIHAASPIIOIHE`

Reading and references

Émile Victor Théodore Myszkowski, *Cryptographie Indéchiffrable basée sur de nouvelles combinaisons rationnelles*, Paris: Société Française d'Imprimerie et de Librairie, 1902, gallica.bnf.fr/ark:/12148/bpt6k1265620p

Helen Fouché Gaines, *Cryptanalysis: a study of ciphers and their solution*, New York: Dover, 1956; previously titled *Elementary Cryptanalysis* and published by American Photographic in 1939; archive.org/details/cryptanalysis00gain; pages 50-51.

American Cryptogram Association, www.cryptogram.org/downloads/aca.info/ciphers/Myszkowski.pdf

Wikipedia, en.wikipedia.org/wiki/Transposition_cipher#Myszkowski_transposition

Merle E. Ohaver, "Solving Cipher Secrets," *Flynn's*, September 17 and 24, 1927, toebes.com/Flynns/Flynns-19270917.htm, toebes.com/Flynns/pdf/Flynns-19270917.pdf, toebes.com/Flynns/Flynns-19270924.htm, toebes.com/Flynns/pdf/Flynns-19270924.pdf

Programming tasks

1. Implement an encryptor for this cipher.
2. Implement a decryptor for this cipher. Be careful about incompletely filled columns.
3. Implement a dictionary attack for a ciphertext that was encrypted with the Myszkowsky cipher.

Exercises

1. Encipher this plaintext with the keyword ABRACADABRA.

   > Here began the experiences that quickly ripened Houdini into the World's Handcuff King and Prison Breaker, which he is, has been and always will be. In exploring his wits for exploits to amuse and entertain the audiences, Houdini hit upon the feat of escaping from ropes tied round him in every conceivable way.
   >
   > (from *The Adventurous Life of a Versatile Artist: Houdini* by Harry Houdini)

2. Decipher this ciphertext with the keyword DOLITTLE.

```
OAYOGHLIEOHALLMHAOKLNNGRTEHHDDWTODTSDDOEMAENSTRWONOOTE
HOOWOCOIAEAHUAARRTCDTEATNSEDTJDTMMSTARRTNEHLNTYWREILRC
ISEIDAPCNEUPEMRSNODFWELEENASRAAMLIHNLEANEWPERAAWT
```

3. Perform a dictionary attack on this ciphertext. The keyword is an English word between five and ten letters in length.

```
WHUHWHTIBEPSIIIYOAHNDNTLHMTBRESOHATENESLYLAHNTNLRSAFBA
TOTATEIGIUESLOTEROLWTATWLDAYSEGTOGMWHGNMICDWBIVIONEOFA
WCCGIERAHKGTKBAYMESTRLROTOMIOHTAOBENOEEEUWBYFTBIOFRTTO
AEERDLVRAEEKNTSKTEOAOEDNCUALHEITUNBEHOFYFERERHTINEHLYB
DTHSSMSTOELIEHEENGFEIEHAIRNSBENMEAIEEIDYLVOATIWAREDHCL
CNTSANITORLTNY
```

Unit 67
Cadenus cipher

The *Cadenus cipher* involves both a columnar transposition and a rotation of the columns. The key is expressed as a keyword in which repeated letters are dropped. The plaintext must be a multiple of 25 times the length of the keyword after repeated letters are removed. These are the steps in encipherment:

1. Divide the plaintext into blocks that are 25 times the length of the keyword (after dropping repeated letters in the key).
2. For each block:
    a. Write the block into a matrix as though for a columnar transposition. Each column has a corresponding letter from the key. Each column has 25 letters.
    b. For each column, roll it downwards by an amount determined by its letter in the key. Use ‘A’ = 0, ‘B’ = 1, ..., ‘V’ = ‘W’ = 21, ..., ‘Z’ = 24 (‘V’ and ‘W’ take the same value so that there are only 25 values).
    c. Apply a columnar transposition using the keyword, without repeated letters.
    d. Read off this block’s part of the ciphertext by rows.

An example seems necessary at this point. We begin with a plaintext that has 375 letters and the keyword `ORATIO` (because Shakespeare did not pronounce his ‘H’s).

```
TO BE OR NOT TO BE THAT IS THE QUESTION WHETHER TIS NOBLER
IN THE MIND TO SUFFER THE SLINGS AND ARROWS OF OUTRAGEOUS
FORTUNE OR TO TAKE ARMS AGAINST A SEA OF TROUBLES AND BY
OPPOSING END THEM TO DIE TO SLEEP NO MORE AND BY A SLEEP TO
SAY WE END THE HEARTACHE AND THE THOUSAND NATURAL SHOCKS
THAT FLESH IS HEIR TO TIS A CONSUMMATION DEVOUTLY TO BE
WISHD TO DIE TO SLEEP TO SLEEP PERCHANCE TO DREAM AY THERES
THE RUB FOR IN THAT SLEEP OF DEATH WHAT DREAMS MAY COME
WUTEVUH
```

After removing repeated letters, our key is `ORATI`, which has five letters. So we break the plaintext into three blocks of 25 × 5 = 125 letters. We lay the first block into a matrix of five columns under the key (shown on the left below). The numbers of steps for rolling each column are ‘O’ = 14, ‘R’ = 17, ‘A’

= 0, ‘T’ = 19, ‘I’ = 8. The columns are rolled downwards (shown in the middle below). Then a columnar transposition is performed (shown on the right).

O	R	A	T	I
T	O	B	E	O
R	N	O	T	T
O	B	E	T	H
A	T	I	S	T
H	E	Q	U	E
S	T	I	O	N
W	H	E	T	H
E	R	T	I	S
N	O	B	L	E
R	I	N	T	H
E	M	I	N	D
T	O	S	U	F
F	E	R	T	H
E	S	L	I	N
G	S	A	N	D
A	R	R	O	W
S	O	F	O	U
T	R	A	G	E
O	U	S	F	O
R	T	U	N	E
O	R	T	O	T
A	K	E	A	R
M	S	A	G	A
I	N	S	T	A
S	E	A	O	F

O	R	A	T	I
T	O	B	T	E
F	I	O	I	O
E	M	E	L	E
G	O	I	T	T
A	E	Q	N	R
S	S	I	U	A
T	S	E	T	A
O	R	T	I	F
R	O	B	N	O
O	R	N	O	T
A	U	I	O	H
M	T	S	G	T
I	R	R	F	E
S	K	L	N	N
T	S	A	O	H
R	N	R	A	S
O	E	F	G	E
A	O	A	T	H
H	N	S	O	D
S	B	U	E	F
W	T	T	T	H
E	E	E	T	N
N	T	A	S	D
R	H	S	U	W
E	R	A	O	U

A	I	O	R	T
B	E	T	O	T
O	O	F	I	I
E	E	E	M	L
I	T	G	O	T
Q	R	A	E	N
I	A	S	S	U
E	A	T	S	T
T	F	O	R	I
B	O	R	O	N
N	T	O	R	O
I	H	A	U	O
S	T	M	T	G
R	E	I	R	F
L	N	S	K	N
A	H	T	S	O
R	S	R	N	A
F	E	O	E	G
A	H	A	O	T
S	D	H	N	O
U	F	S	B	E
T	H	W	T	T
E	N	E	E	T
A	D	N	T	S
S	W	R	H	U
A	U	E	R	O

We read off the ciphertext by rows. The remaining blocks are processed in the same way, and in the end we get this ciphertext:

```
BETOTOOFIIEEEMLITGOTQRAENIASSUEATSTTFORIBORONNTOROIHAUOSTMT
GREIRFLNSKNAHTSORSRNAFEOEGAHAOTSDHNOUFSBETHWTTENEETADNTSSWR
HUAUEROOSEOTSAAAEYLNAOSKHPDNTAYLMHNDOEITEEEIACHMBTDRNNSHESP
SNHTNFUUWTIHNTORTAAOTLCHPLSATRDTSOBPREDEGETRSHBUOEDOAHESEOE
TNEIHAEIDOEYLTCSAEDMOOLSOTMETOTENETMHDCICFAEOHTREOVLETPNFOS
EAHHURDDEICTSDTRBOWPYHARAEISMARLSMAROTTYAPEAWEHLUVETEIOHADV
MEHEYNATEWUEBSTOENPTS
```

Reading and references

American Cryptogram Association, www.cryptogram.org/downloads/aca.info/ciphers/Cadenus.pdf

Programming tasks

1. Implement an encryptor.
2. Implement a decryptor.
3. Implement a dictionary attack.

Exercises

1. Encipher this text with the keyword `INTERNET`.

   > While my internet is down, I have to write my own plaintexts. It is very frustrating, but this is the cost we bear when we steal internet from the store down the street. I'm sure you don't want to hear about that. Perhaps you would like some historical bits of information about the Cadenus cipher? Unfortunately, I don't know any. Let me look some up online...

2. Decipher this text with the keyword `HEART`.

   ```
   DARHNAIIRTNRAEIWEDTLOAWANJSERHHAHTNIACUIRPUHWIYVTBITSE
   ESWAIEVOBODMGACNIRSHRPEFIESRSXHFOFESHHDNXDSEENCIHADNAG
   ESTSAOSWELAMSCDFN
   ```

3. Break this ciphertext with a dictionary attack.

   ```
   EWOIETEHTDOEINPEHAEAAERRRDNWIOTDHYHLEDONRPWTEPWSRFMNAE
   HHHEHNLQAORRGECAXEUHLHREHRSATNRASEESRWSLITAMITAPIIEPSI
   SAHWPWEWJOPEIDWUEDDSNATSSTTANTOARATEAAGLTISAIRNGENBUUI
   NFSEEFOXIODWATGNOEAFESEHDEIITIRDYSGSOGRENCBNIRUDLCSNTO
   AGBATTOITALNHNIGIIPLVDESHEHTTVIIHEUDTNTNPTAINSEPTPWEOS
   HETLWESLEEWAMHWBSLIRNCRRUUTTCSGRHEEYTLLMROMERENCOTHATE
   GNFHWAAIDGINUFYNRSNSAMORAUGSWBNTIFEREDILLTILEOEAERSHEY
   NTDDNEVIRRNSBXSTXWHERDGVITISNAMEPSOSHBHWWBWDTERTEEATAE
   DTSSLSTTRIATESUOEOSELOEENSNWEANEOAATASHAIRPPANWEOSIEHO
   NLCHNNTUDTISEE
   ```

Unit 68
Hill-climbing attack on the Cadenus cipher

In this attack we unlink the rolling and columnar transposition and vary the keys of the two operations separately; at the end, we put them back together. Therefore, we need a decryptor that applies them separately. Before we apply the attack, we have to determine the key length (or guess it). In the exercises in Unit 9 we found a cutoff above which we are confident that we have English text; we will use that cutoff in the algorithm. We will be using the usual technique of working with parent and child keys. To avoid local maxima, we also need to use a margin of error. For this cipher, we will vary this margin so that it vanishes when we reach the fitness cutoff.

0. set the alphabet to be `ABCDERGHIJKLMNOPQRSTUVXYZ` (no ‘W’)
1. initialize the parent shift key to an array of 0s
2. initialize the parent permutation key to (0 1 2 ...)
3. calculate the best fitness as the fitness of the unmodified ciphertext
4. set a counter to 0
5. while counter is less than 1,000
    a. copy the parent shift key into a child shift key
    b. copy the parent permutation to a child permutation
    c. randomly choose one of these ways to modify the child keys:
        i. change one member of the child shift key to a random number in 0, ..., 25
        ii. swap two randomly selected members of the child permutation
        iii.
            - randomly choose a number *n* from 1 to the key length
            - roll both keys leftward *n* of steps (with rollover)
            - subtract 1 from each of the last *n* members of the child shift key
        iv.
            - randomly choose a number *n* from 1 to 25
            - add *n* to each member of the child shift key, modulo 25
    d. decipher the ciphertext with the child keys
    e. calculate the new fitness of the new plaintext
    f. set the margin to be
        i. 0 if the new fitness exceeds the cutoff
        ii. the square root of the cutoff minus the new fitness, all divided by 10,

if the new fitness is less than the cutoff

g. if (the new fitness exceeds the best fitness) or
((the new fitness exceeds the best fitness minus the margin) and
(we roll a 1 on a 20-sided die))
    i. copy the child shift key into the parent shift key
    ii. copy the child permutation into the parent permutation
    iii. set the best fitness equal to the new fitness
    iv. set the counter to 0

h. increment the counter

6. convert the parent shift key to a keyword; for each member of the shift key:
    a. take *n* to be 25 minus the shift
    b. the letter of the keyword is the n^{th} letter of the alphabet (without 'W')
7. output the keyword

Programming tasks

1. Write a function that deciphers a ciphertext with separate keys for the shifts and for the permutation.

2. Implement the attack. Use your function from Exercise 1, and use tetragram fitness.

Exercises

1. Use your attack to break the example from the previous unit.

2. Break this ciphertext from the 2014 British National Cipher Challenge. It may take several attempts.

```
AFCAEUOTTACTHRIOLETCSERTHSHTRAHKYORPFRGEOADPPJNGLTERNE
FEOFIORTSDDOEEUMSCRUERNFETLAAFSTWIENTRVOONERHUAHRAVERE
ETSVSIELHLOSTDOALOYAESMNNDIGNNRHOHHTSNAOILNCNSSICREANN
EEIIIERWTANESRVOGIEIYWSSDGPVOIAISAOAEOAEDRNITRNXEIGRPS
SHADHDTOIPAATEXENNESAGROBTLESRNROIRYPBGEDCLLIWALALEENI
GRRNWYRLIMLPSTOLEFTRDMUARIEEEIIAOLNEWSAOHRTLSTOBETNSLV
FIVDOVTPOAEEISCIOHIPSEVEEDTEWFARNHEBLEAOTOHTTTEPNCKAON
HWETMVYPRREONNASGDEDOEEEOAAMTCICTTIFNADRESRTSEROSETRHC
ICTPSAAEHLDHSFXSOAOTCTBBSOEIRNSADLYTRRUNRCEPTTHREUHNKT
ACECEELRWNIREEEAESEEEIDISOGCEOMNRTEJHAGABSENITLWTRNBMI
ELSARETESRNGSNHEBIOSDIENAFLEISAHOCIFEVMFATANATRNIAGNHA
TNMIBNIUFENRTOTTRNYPAIDYIEGDNMERHHIOTRETCESSEILDRBCEPR
IGAESOADLTAHIEVEBRCENLEVASADNNTHNEITEIISAHUHHUAMONEFYH
LONWHAEEEEOSNEEYANEISETOGYITERLIHTCMIOIRARFDOETNIHTNEH
IIKAMRDMNADANAODSESEIYCLSIANTAOLTCIYMIDENTTHLTNDXTTTMA
SBLEAEETLISIRTWTURPFAILTEAOEFEISIIIYISIKVTWISPRBSINELP
```

HRMOHIAGNLSLVITODAISDPNYDDCAAOTAHCEHTUEIRREDAECTOSNRHV
NAODOIKOETCINENEURRISDCOURAGLVIMMUPPDITEANDITMAAIAIELE
ONNREEDAODBOIUMELROTNTTTGITNRLRIENNIKLYSOGSTCIFYPIPVID
VSSMNCEIASIITSNNEATITOMRHBNHNIDPRLREPOYNALSNVSDOSANESI
TFAENLTGODATTEEAISICROOTMSMFHAUENIRSGHYNWEINTEGODIILEE
DTARNOSRCAAENDTCUTTFDRBEHTMFITOORDRUIAOYAANOEELDOINHUS
GITEAORIECEVEMNTRATMTFPEUCUTAHAMTNEWONICDEEMRPAOLITOAF
ESOOSSPFNLNEEOOTACHLLIRSSXSOFPDFTFRNPRAEEAYLONAHAUTNTC
NTCBAWLONEFTOATECVOWDLWVNNEEDTIIOIGTEGMTAHEEATEFAAEPRR
CROSHEERRPALEDIENGIDRREOUHVESUROYTNSOSINUIUIOFPRDA