**Повторяем теорию: работа со строками и символами**

string.ascii\_lowercase = 'abcdefghijklmnopqrstuvwxyz'

string.ascii\_uppercase = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'

str.**find**(substr) - ищет в строке str первое вхождение подстроки substr и возвращает индекс начала подстроки. Если не находит, возвращает -1.

>>> a = list('Hello')

>>> print(a)

['H', 'e', 'l', 'l', 'o']

>>> s = '-'.join(a)

>>> s

'H-e-l-l-o'

>>> ord('a')

97

>>> chr(97)

'a'

>>> 'HElLo, WoRlD'.lower()

'hello, world'

>>> 'HElLo, WoRlD'.upper()

'HELLO, WORLD'

Slice - часть строки или списка [от:до] или [от:до:шаг]

>>> s = 'Hello!'

>>> s[1:4]

'ell'

>>> s[2:]

'llo!'

>>> s[:2]

'He'

>>> s[1:5:2]

'el'

>>> s[0:5:2]

'Hlo'

>>> s[::2]

'Hlo'

>>> s[::-2]

'!le'

>>> s[::-1]

'!olleH'

Перебираем с индексами:

for i, c in enumerate('hello'):

print(i, c)

# получим

# 0 h

# 1 e

# 2 l

# 3 l

# 4 o

[Повторяем списки](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonList)

**Шифры**

Переписку могут перехватывать. Чтобы письма не могли читать другие люди, их шифровали.

**Шифр Цезаря**

Очень старый шифр. Алфавит сдвигается на N букв и получается зашифрованный алфавит. Cдвинем на 3 буквы, получим:

abcdefghijklmnopqrstuvwxyz - что было

defghijklmnopqrstuvwxyzabc - зашифрованный алфавит

Зашифруем текст, заменяя букву а на d, b на e, с на f и так далее.

Строка 'hello' превратится в 'khoor'.

**Задача 1.**

Написать одному человеку кодирование (шифрование), а другому - декодирование (расшифровку).

Шифрование:

|  |  |
| --- | --- |
| [**Input**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=0&table=1&up=0#sorted_table) | [**Output**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=1&table=1&up=0#sorted_table) |
| 3 hello | khoor |
| 3 hello, world! | khoor, zruog! |

Расшифровка:

|  |  |
| --- | --- |
| [**Input**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=0&table=2&up=0#sorted_table) | [**Output**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=1&table=2&up=0#sorted_table) |
| -3 khoor | hello |
| -3 khoor, zruog! | hello, world! |

n = int(input())

text = input()

text = text.lower() # все буквы маленькие

oldalphabet = 'abcdefghijklmnopqrstuvwxyz'

# или можно написать так:

# oldalphabet = string.ascii\_lowercase

def make\_alphabet(oldalphabet, n):

"""

из алфавита oldalphabet делает новый алфавит, сдвигая его на n букв.

Возвращает новый алфавит

"""

# тут нужно написать код

def code\_subst(oldalphabet, newalphabet, text):

"""

Шифрует подстановками text из алфавита oldalphabet в алфавит newalphabet.

Возвращает зашифрованный текст

"""

# тут нужно написать код

newalphabet = make\_alphabet(oldalphabet, n)

codetext = code\_subst(oldalphabet, newalphabet, text)

print(codetext)

**Задача 2. Взлом шифра**

Вы знаете, что фраза зашифрована кодом цезаря с неизвестным сдвигом. Попробуйте все возможные сдвиги и расшифруйте фразу.

Номер варианта дает преподаватель

1 zcysrgdsj gq zcrrcp rfyl sejw.

2 ibtpmgmx mw fixxiv xler mqtpmgmx.

3 rhlokd hr adssdq sgzm bnlokdw.

4 tfdgcvo zj svkkvi kyre tfdgcztrkvu.

5 xdsl ak twllwj lzsf fwklwv.

6 wtevwi mw fixxiv xler hirwi.

7 cplolmtwtej nzfyed.

8 qncagyj ayqcq ypcl'r qncagyj clmsef rm zpcyi rfc psjcq.

9 itbpwcop xzikbqkitqbg jmiba xczqbg.

10 uhhehi ixekbt duluh fqii iybudjbo.

11 sljcqq cvnjgagrjw qgjclacb.

12 sx dro pkmo yp kwlsqesdi, bopeco dro dowzdkdsyx dy qeocc.

13 lzwjw kzgmdv tw gfw-- sfv hjwxwjstdq gfdq gfw --gtnagmk osq lg vg al.

14 grznuamn zngz cge sge tuz hk uhbouay gz loxyz atrkyy eua'xk jazin.

15 dem yi rujjuh jxqd duluh.

16 fqymtzlm sjajw nx tkyjs gjyyjw ymfs \*wnlmy\* stb.

17 fc qeb fjmibjbkqxqflk fp exoa ql bumixfk, fq'p x yxa fabx.

18 xu iwt xbeatbtcipixdc xh tphn id tmeapxc, xi bpn qt p vdds xstp.

19 zmyqebmoqe mdq azq tazwuzs sdqmf upqm — xqf'e pa yadq ar ftaeq!

**Шифр Цезаря с изменением алфавита**

Чтобы шифр было труднее взломать, новый алфавит делается с ключевой фразой.

Буквы из ключевой фразы берутся по 1 разу, остальные буквы берутся из алфавита.

abcdefghijklmnopqrstuvwxyz + apple

bcdfghijkmnoqrstuvwxyzaple

**Задача 3.1: буквы в числа**

Дан текст. Замените каждую букву на число - порядковый номер буквы в алфавите. А на 1, В на 2, С на 3 и так далее.

Пробел заменять на 0. Другие символы оставлять как есть.

Числа печатайте через пробел.

|  |  |
| --- | --- |
| [**Input**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=0&table=3&up=0#sorted_table) | [**Output**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=1&table=3&up=0#sorted_table) |
| hello | 8 5 12 12 15 |
| Hello, world! | 8 5 12 12 15 , 0 23 15 18 12 4 ! |

**Задача 3.2: Обратная задача**

Замените числа, написанные через пробел, на буквы. Не числа не изменять.

|  |  |
| --- | --- |
| [**Input**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=0&table=4&up=0#sorted_table) | [**Output**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=1&table=4&up=0#sorted_table) |
| 8 5 12 12 15 | hello |
| 8 5 12 12 15 , 0 23 15 18 12 4 ! | hello, world! |

**Задача 3.3: Убрать гласные буквы**

В английском алфавите буквы eyuioa - гласные. Удалите из текста все гласные буквы.

|  |  |
| --- | --- |
| [**Input**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=0&table=5&up=0#sorted_table) | [**Output**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=1&table=5&up=0#sorted_table) |
| apple | ppl |
| Good morning! | Gd mrnng! |

**Задача 3.4: Оставить только НЕ гласные буквы**

В английском алфавите буквы eyuioa - гласные. Удалите из текста все гласные буквы. Удалите из текста все НЕ буквы.

|  |  |
| --- | --- |
| [**Input**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=0&table=6&up=0#sorted_table) | [**Output**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=1&table=6&up=0#sorted_table) |
| apple | ppl |
| Good morning! | Gdmrnng |

**Задача 3.5: Убрать повторяющиеся буквы и лишние символы**

Построим по ключевой фразе часть алфавита. Возьмем все буквы по одному разу. Не буквы убрать.

Буквы должны идти в том порядке, в котором встретились в первый раз.

|  |  |
| --- | --- |
| [**Input**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=0&table=7&up=0#sorted_table) | [**Output**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=1&table=7&up=0#sorted_table) |
| apple | aple |
| Good morning! | godmrni |

**Задача 3.6: Убрать из первого слова все буквы, которые есть во втором слове**

Дан алфавит (все буквы уникальные) и сжатое ключевое слово (каждая буква встречается только 1 раз).

Построить новый алфавит, который состоит из сжатого ключевого слова и оставшихся букв алфавита.

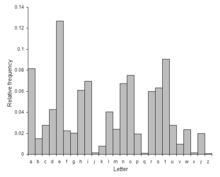
|  |  |
| --- | --- |
| [**Input**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=0&table=8&up=0#sorted_table) | [**Output**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=1&table=8&up=0#sorted_table) |
| apple abcdefghijklmnopqrstuvwxyz | aplebcdfghijkmnoqrstuvwxyz |
| godmrni abcdefghijklmnopqrstuvwxyz | godmrniabcefhjklpqstuvwxyz |

**Задача 3.7: Убрать повторяющиеся буквы и лишние символы**

Зашифруйте текст. Ключевой фразой возьмите свое имя. Сдвиг 3.

**Устойчивость шрифтов с подстановками**

Шрифты с подстановками легко взломать, зная как часто встречаются буквы в тексте.



Видно, что всего в английском языке встречается буква e.

**Бинарный шифр**

Поставим каждому символу свое число (по порядку)

Число в десятичной системе представляется как 495 = 4\*102 + 9\*101 + 5\*102

Числа можно записывать как суммы степеней числа 2, а не 10. Это бинарное число.

10112 = 1\*23 + 0\*22 + 1\*21 + 1\*20 = 1\*8 + 0\*4 + 1\*2 + 1\*1 = 8+4+1 = 1310

пробел - 0 - 00000

a - 1 - 00001

b - 2 - 00010

c - 3 - 00011

d - 4 - 00100

e - 5 - 00101

f - 6 - 00110

g - 7 - 00111

h - 8 - 01000

i - 9 - 01001

j - 10 - 01010

k - 11 - 01011

l - 12 - 01100

m - 13 - 01101

n - 14 - 01110

o - 15 - 01111

p - 16 - 10000

q - 17 - 10001

r - 18 - 10010

s - 19 - 10011

t - 20 - 10100

u - 21 - 10101

v - 22 - 10110

w - 23 - 10111

x - 24 - 11000

y - 25 - 11001

z - 26 - 11010

Хотим зашифровать слово hello. Это числа

h - 8 - 01000

e - 5 - 00101

l - 12 - 01100

l - 12 - 01100

o - 15 - 01111

Выпишем бинарные числа.

01000 00101 01100 01100 01111

0 - маленькая буква, 1 - большая буква.

Зашифруем эту последовательность из 0 и 1 в тексте

Beautiful is better than ugly. Explicit is better than implicit. Simple is better than complex.

01000 00101 01100 01100 01111

010000010 10 110001 1000 1111

beautiful is better than ugly. explicit is better than implicit.

bEautifUl Is BEtteR[?](http://acm.mipt.ru/twiki/bin/edit/Cintro/BEtteR?topicparent=Cintro.PythonCodes) Than UGLY

**Задача 4.0: Буквы в числа**

Из строки получите список чисел.

Каждую букву замените на ее порядковый номер. А на 1, В на 2, С на 3 и так далее.

Пробел заменить на 0. Прочие символы удалить.

|  |  |
| --- | --- |
| [**Input**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=0&table=9&up=0#sorted_table) | [**Output**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=1&table=9&up=0#sorted_table) |
| hello | [8, 5, 12, 12, 15] |
| Hello, world! | [8, 5, 12, 12, 15, 0, 23, 15, 18, 12, 4] |

**Задача 4.1: Цифры числа**

Напечатайте 4 последних цифры числа в обратном порядке.

|  |  |
| --- | --- |
| [**Input**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=0&table=10&up=0#sorted_table) | [**Output**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=1&table=10&up=0#sorted_table) |
| 1279 | [9, 7, 2, 1] |
| 9876543206 | [6, 0, 2, 3] |
| 15 | [5, 1, 0, 0] |

**Задача 4.2: Цифры числа в прямом порядке**

Напечатайте 4 последних цифры числа в обратном порядке.

|  |  |
| --- | --- |
| [**Input**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=0&table=11&up=0#sorted_table) | [**Output**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=1&table=11&up=0#sorted_table) |
| 1279 | [1, 2, 7, 9] |
| 9876543206 | [3, 2, 0, 6] |
| 15 | [0, 0, 1, 5] |

**Задача 4.3: Бинарное представление числа**

Дано число. Напечатайте его 6 последних бинарных цифр.

|  |  |
| --- | --- |
| [**Input**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=0&table=12&up=0#sorted_table) | [**Output**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=1&table=12&up=0#sorted_table) |
| 5 | [0, 0, 1, 0, 1] |
| 1 | [0, 0, 0, 0, 1] |
|  | [0, 0, 0, 0, 0] |
| 15 | [0, 1, 1, 1, 1] |

**Задача 4.4: НеРоВнЫй[?](http://acm.mipt.ru/twiki/bin/edit/Cintro/%D0%9D%D0%B5%D0%A0%D0%BE%D0%92%D0%BD%D0%AB%D0%B9?topicparent=Cintro.PythonCodes) ПоЧеРк[?](http://acm.mipt.ru/twiki/bin/edit/Cintro/%D0%9F%D0%BE%D0%A7%D0%B5%D0%A0%D0%BA?topicparent=Cintro.PythonCodes)**

В тексте напишите буквы в порядке "большая буква", "маленькая буква". Не буквы оставлять как есть. Они на порядок букв не влияют.

|  |  |
| --- | --- |
| [**Input**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=0&table=13&up=0#sorted_table) | [**Output**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonCodes?sortcol=1&table=13&up=0#sorted_table) |
| hello | HeLlO[?](http://acm.mipt.ru/twiki/bin/edit/Cintro/HeLlO?topicparent=Cintro.PythonCodes) |
| a.b,c | A.b,C |
| Hello, Harry Potter! | HeLlO[?](http://acm.mipt.ru/twiki/bin/edit/Cintro/HeLlO?topicparent=Cintro.PythonCodes), HaRrY[?](http://acm.mipt.ru/twiki/bin/edit/Cintro/HaRrY?topicparent=Cintro.PythonCodes) pOtTeR! |

**Задача 4.5: НеРоВнЫй[?](http://acm.mipt.ru/twiki/bin/edit/Cintro/%D0%9D%D0%B5%D0%A0%D0%BE%D0%92%D0%BD%D0%AB%D0%B9?topicparent=Cintro.PythonCodes) ПоЧеРк[?](http://acm.mipt.ru/twiki/bin/edit/Cintro/%D0%9F%D0%BE%D0%A7%D0%B5%D0%A0%D0%BA?topicparent=Cintro.PythonCodes) по образцу**

Дан список из 0 и 1 - образец.

Дана строка. В строке букв больше, чем чисел в образце.

Буквы в строке должны стать большими и маленькими в таком порядке, который указан в образце. Если в списке очередное число 0, то буква маленькая. Если в списке очередное число 1, то буква большая.

Когда образец (список чисел) закончится,

case = [0, 1, 1, 0, 1, 0, 0, 1]

text = 'Hello, world!'

def encode(text, case):

# тут нужно написать код

result = encode(text, case)

print(result) # hELlO, woR

**Задача 4.6: Бинарный шифр.**

Напишите программу, которая шифрует ваше имя в этом тексте:

Beautiful is better than ugly.

Explicit is better than implicit.

Simple is better than complex.

Complex is better than complicated.

Flat is better than nested.

Sparse is better than dense.

Readability counts.

Special cases aren't special enough to break the rules.

Although practicality beats purity.

Errors should never pass silently.

Unless explicitly silenced.

In the face of ambiguity, refuse the temptation to guess.

There should be one-- and preferably only one --obvious way to do it.

Although that way may not be obvious at first unless you're Dutch.

Now is better than never.

Although never is often better than \*right\* now.

If the implementation is hard to explain, it's a bad idea.

If the implementation is easy to explain, it may be a good idea.

Namespaces are one honking great idea — let's do more of those!