

# Programmazione in Python

Esercizi

Dario Pescini - Mirko Cesarini <sup>1</sup>

Università degli Studi di Milano-Bicocca

Dipartimento di Statistica e Metodi Quantitativi

`dario.pescini@unimib.it`

# Indovina la parola

Si chiede di implementare un programma che, scelta una parola a caso tra quelle presenti nel file `listaParoleIta.txt`, chieda all'utente di indovinarla.

Il programma dovrà, inoltre, fornire come aiuto il conteggio, per ciascuno dei caratteri presenti nella parola proposta dall'utente, delle sue occorrenze nella parola da indovinare.

```
ossequiosi
```

```
la parola è composta da 12 caratteri  
che parola è? (-1 per terminare) computer  
aiuto:
```

```
c e' presente 0 volte  
e e' presente 1 volte  
m e' presente 0 volte  
o e' presente 2 volte  
p e' presente 0 volte  
r e' presente 0 volte  
u e' presente 1 volte  
t e' presente 0 volte
```

## file listaParoleIta.txt

```
a  
abate  
abati  
abbagli  
abbaglia  
abbagliai  
abbagliamo  
abbagliano  
abbagliare  
abbagliata  
abbagliate  
abbagliati  
  
....  
  
zuffoli  
zuffoliamo  
zuffolino  
zuffolo  
zuppa  
zuppe  
zuppiera  
zuzzurellone
```

# Soluzione

```
import random
fileName = 'listaParoleIta.txt'

# caricamento lista parole possibili target
fileParole = open(fileName, 'r')
listaParole = fileParole.readlines()
fileParole.close()
lunghezzaLista = len(listaParole)

# sorteggio della parola segreta
iParola = random.randint(0, lunghezzaLista - 1)
parolaDaIndovinare = listaParole[iParola]
# print parolaDaIndovinare
print 'la parola è composta da ', len(parolaDaIndovinare), 'caratteri'
```

```
# inserimento tentativi e controllo soluzione
termina = False
indovinato = False
while termina == False and indovinato == False:
    parolaTentativo = raw_input('che parola è? (-1 per terminare) ')
    if parolaTentativo == '-1':
        termina = True
    else:
        if parolaTentativo == parolaDaIndovinare:
            print 'hai indovinato'
            indovinato = True
        else:
            conteggio = {}
            for carattere in parolaTentativo:
                if carattere not in conteggio.keys():
                    conteggio[carattere] =
parolaDaIndovinare.count(carattere)
            print "aiuto:"
            for el in conteggio.keys():
                print el, "e' presente ", conteggio[el], 'volte'
```

# Codice Fiscale

Si chiede di implementare in python l'algoritmo usato per la generazione del codice fiscale.

I primi quindici caratteri sono indicativi dei dati anagrafici di ciascun soggetto secondo l'ordine seguente:

- tre caratteri alfabetici per il **cognome**;
- tre caratteri alfabetici per il **nome**;
- due caratteri numerici per l'**anno di nascita**;
- un carattere alfabetico per il **mele di nascita**;
- due caratteri numerici per il **giorno di nascita** ed il **semo**;
- quattro caratteri, di cui uno alfabetico e tre numerici per il **comune italiano** o per lo Stato estero di nascita.

Il sedicesimo carattere, alfabetico, ha funzione di **controllo**.

# Generatore codice fiscale: caratteri indicativi del cognome

- I cognomi che risultano composti da più parti o comunque separati od interrotti, vengono considerati come se fossero scritti secondo un'unica ed ininterrotta successione di caratteri.
- Per i soggetti coniugati di sesso femminile si prende in considerazione soltanto il cognome da nubile.
- Se il cognome contiene tre o più consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima, la seconda e la terza consonante.
- Se il cognome contiene due consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima e la seconda consonante e la prima vocale.
- Se il cognome contiene una consonante e due vocali, si rilevano, nell'ordine, quella consonante e quindi la prima e la seconda vocale.
- Se il cognome contiene una consonante e una vocale, si rilevano la consonante e la vocale, nell'ordine, e si assume come terzo carattere la lettera x (ics).
- Se il cognome è costituito da due sole vocali, esse si rilevano, nell'ordine, e si assume come terzo carattere la lettera x (ics).

# Generatore codice fiscale: caratteri indicativi del nome

- I nomi doppi, multipli o comunque composti, vengono considerati come scritti per esteso in ogni loro parte e secondo un'unica ed ininterrotta successione di caratteri.
- Se il nome contiene quattro o più consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima, la terza e la quarta consonante.
- Se il nome contiene tre consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima, la seconda e la terza consonante.
- Se il nome contiene due consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima e la seconda consonante e la prima vocale.
- Se il nome contiene una consonante e due vocali, i tre caratteri da rilevare sono, nell'ordine, quella consonante e quindi la prima e la seconda vocale.
- Se il nome contiene una consonante e una vocale, si rilevano la consonante e la vocale, nell'ordine, e si assume come terzo carattere la lettera x (ics).
- Se il nome è costituito da due sole vocali, esse si rilevano nell'ordine, e si assume come terzo carattere la lettera x (ics).



## Generatore codice fiscale: Data, sesso e luogo di nascita

- I due caratteri numerici indicativi dell'anno di nascita sono, nell'ordine, la cifra delle decine e la cifra delle unità dell'anno stesso.
- Il carattere alfabetico corrispondente al mese di nascita e' quello stabilito per ciascun mese nella seguente tabella:

Gennaio = A	Maggio = E	Settembre = P
Febbraio = B	Giugno = H	Ottobre = R
Marzo = C	Luglio = L	Novembre = S
Aprile = D	Agosto = M	Dicembre = T

# Generatore codice fiscale: Data, sesso e luogo di nascita

- I due caratteri numerici indicativi del giorno di nascita e del sesso vengono determinati nel modo seguente:
  - Per i soggetti maschili il giorno di nascita figura invariato, con i numeri da uno a trentuno, facendo precedere dalla cifra zero i giorni del mese dall'uno al nove.
  - Per i soggetti femminili il giorno di nascita viene aumentato di quaranta unità, per cui esso figura con i numeri da quarantuno a settantuno.
- I quattro caratteri alfanumerici indicativi del comune italiano o dello Stato estero di nascita, sono costituiti da un carattere alfabetico seguito da tre caratteri numerici, secondo la codifica stabilita dall'Agenzia del Territorio e contenuta nel file `CodiceComuniItalia.csv`

# Generatore codice fiscale: Carattere alfabetico di controllo

Il sedicesimo carattere ha funzione di controllo dell'esatta trascrizione dei primi quindici caratteri e viene determinato in questo modo:

- ciascuno degli anzidetti quindici caratteri, a seconda che occupi posizione di ordine pari o posizione di ordine dispari, viene convertito in un valore numerico, in base alle tabelle di corrispondenza successivamente riportate...

Per i sette caratteri con posizione di ordine

- pari: file `CFTabellaPari.tsv`
- dispari: file `CFTabellaDispari.tsv`

## Generatore codice fiscale: Carattere alfabetico di controllo

- I valori numerici così determinati vengono addizionati e la somma si divide per il numero 26.
- Il carattere di controllo si ottiene convertendo il resto di tale divisione nel carattere alfabetico ad esso corrispondente nel file `CFTabellaResto.tsv`

```
nome = 'alessandro'  
cognome = 'manzoni'  
dataNascita = '07031785'  
genere = 'M'  
comune = 'milano'
```

```
MNZLSN85C07F205D
```

# Cognome

- I cognomi che risultano composti da più parti o comunque separati od interrotti, vengono considerati come se fossero scritti secondo un'unica ed ininterrotta successione di caratteri.
- Per i soggetti coniugati di sesso femminile si prende in considerazione soltanto il cognome da nubile.
- Se il cognome contiene tre o più consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima, la seconda e la terza consonante.
- Se il cognome contiene due consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima e la seconda consonante e la prima vocale.
- Se il cognome contiene una consonante e due vocali, si rilevano, nell'ordine, quella consonante e quindi la prima e la seconda vocale.
- Se il cognome contiene una consonante e una vocale, si rilevano la consonante e la vocale, nell'ordine, e si assume come terzo carattere la lettera x (ics).
- Se il cognome e' costituito da due sole vocali, esse si rilevano, nell'ordine, e si assume come terzo carattere la lettera x (ics).

# Cognome

- I cognomi che risultano composti da più parti o comunque separati od interrotti, vengono considerati come se fossero scritti secondo un'unica ed ininterrotta successione di caratteri.
- Per i soggetti coniugati di sesso femminile si prende in considerazione soltanto il cognome da nubile.
- Se il cognome contiene tre o più consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima, la seconda e la terza consonante.
- Se il cognome contiene due consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima e la seconda consonante e la prima vocale.
- Se il cognome contiene una consonante e due vocali, si rilevano, nell'ordine, quella consonante e quindi la prima e la seconda vocale.
- Se il cognome contiene una consonante e una vocale, si rilevano la consonante e la vocale, nell'ordine, e si assume come terzo carattere la lettera x (ics).
- Se il cognome e' costituito da due sole vocali, esse si rilevano, nell'ordine, e si assume come terzo carattere la lettera x (ics).

consonanti + vocali + x

# Cognome

- I cognomi che risultano composti da più parti o comunque separati od interrotti, vengono considerati come se fossero scritti secondo un'unica ed ininterrotta successione di caratteri.
- Per i soggetti coniugati di sesso femminile si prende in considerazione soltanto il cognome da nubile.
- Se il cognome contiene tre o più consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima, la seconda e la terza consonante.
- Se il cognome contiene due consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima e la seconda consonante e la prima vocale.
- Se il cognome contiene una consonante e due vocali, si rilevano, nell'ordine, quella consonante e quindi la prima e la seconda vocale.
- Se il cognome contiene una consonante e una vocale, si rilevano la consonante e la vocale, nell'ordine, e si assume come terzo carattere la lettera x (ics).
- Se il cognome e' costituito da due sole vocali, esse si rilevano, nell'ordine, e si assume come terzo carattere la lettera x (ics).

consonanti + vocali + x  
stringa[:3]



# Cognome

```
def estraiCarCognome(parola):  
    parola = parola.replace(' ', '')  
    parola = parola.upper()  
    frammentoConsonanti = ''  
    frammentoVocali = ''  
    for carattere in parola:  
        if carattere not in ['A', 'E', 'I', 'O', 'U']:  
            frammentoConsonanti += carattere  
        else:  
            frammentoVocali += carattere  
    parolaRiordinata = frammentoConsonanti + frammentoVocali +  
    'XXX'  
    # print parolaRiordinata  
    return parolaRiordinata[:3]
```

# Nome

- I nomi doppi, multipli o comunque composti, vengono considerati come scritti per esteso in ogni loro parte e secondo un'unica ed ininterrotta successione di caratteri.
- Se il nome contiene quattro o più consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima, la terza e la quarta consonante.
- Se il nome contiene tre consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima, la seconda e la terza consonante.
- Se il nome contiene due consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima e la seconda consonante e la prima vocale.
- Se il nome contiene una consonante e due vocali, i tre caratteri da rilevare sono, nell'ordine, quella consonante e quindi la prima e la seconda vocale.
- Se il nome contiene una consonante e una vocale, si rilevano la consonante e la vocale, nell'ordine, e si assume come terzo carattere la lettera x (ics).
- Se il nome è costituito da due sole vocali, esse si rilevano nell'ordine, e si assume come terzo carattere la lettera x (ics).

# Nome

- I nomi doppi, multipli o comunque composti, vengono considerati come scritti per esteso in ogni loro parte e secondo un'unica ed ininterrotta successione di caratteri.
- Se il nome contiene quattro o più consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima, la terza e la quarta consonante.
- Se il nome contiene tre consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima, la seconda e la terza consonante.
- Se il nome contiene due consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima e la seconda consonante e la prima vocale.
- Se il nome contiene una consonante e due vocali, i tre caratteri da rilevare sono, nell'ordine, quella consonante e quindi la prima e la seconda vocale.
- Se il nome contiene una consonante e una vocale, si rilevano la consonante e la vocale, nell'ordine, e si assume come terzo carattere la lettera x (ics).
- Se il nome è costituito da due sole vocali, esse si rilevano nell'ordine, e si assume come terzo carattere la lettera x (ics).

consonanti + vocali + x

# Nome

- I nomi doppi, multipli o comunque composti, vengono considerati come scritti per esteso in ogni loro parte e secondo un'unica ed ininterrotta successione di caratteri.
- Se il nome contiene quattro o più consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima, la terza e la quarta consonante.
- Se il nome contiene tre consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima, la seconda e la terza consonante.
- Se il nome contiene due consonanti, i tre caratteri da rilevare sono, nell'ordine, la prima e la seconda consonante e la prima vocale.
- Se il nome contiene una consonante e due vocali, i tre caratteri da rilevare sono, nell'ordine, quella consonante e quindi la prima e la seconda vocale.
- Se il nome contiene una consonante e una vocale, si rilevano la consonante e la vocale, nell'ordine, e si assume come terzo carattere la lettera x (ics).
- Se il nome è costituito da due sole vocali, esse si rilevano nell'ordine, e si assume come terzo carattere la lettera x (ics).

consonanti + vocali + x  
stringa[1,3,4] or stringa[:3]

# Nome

```
def estraiCarNome(parola):  
    parola = parola.replace(' ', '')  
    parola = parola.upper()  
    frammentoConsonanti = ''  
    frammentoVocali = ''  
    for carattere in parola:  
        if carattere not in ['A', 'E', 'I', 'O', 'U']:  
            frammentoConsonanti += carattere  
        else:  
            frammentoVocali += carattere  
    if len(frammentoConsonanti) > 3:  
        frammentoConsonanti = frammentoConsonanti[  
            0] + frammentoConsonanti[2] + frammentoConsonanti[3]  
    parolaRiordinata = frammentoConsonanti + frammentoVocali +  
        'XXX'  
    # print parolaRiordinata  
    return parolaRiordinata[:3]
```

# Data

- I due caratteri numerici indicativi dell'**anno di nascita** sono, nell'ordine, la cifra delle **decine** e la cifra delle **unità** dell'anno stesso.
- Il carattere alfabetico corrispondente al  **mese di nascita** e' quello stabilito per ciascun mese nella seguente tabella:

Gennaio = A	Maggio = E	Settembre = P
Febbraio = B	Giugno = H	Ottobre = R
Marzo = C	Luglio = L	Novembre = S
Aprile = D	Agosto = M	Dicembre = T

# Data

```
def estraiCarMese(parola):  
    dizionarioMesi = {'01': 'A', '02': 'B', '03': 'C', '04': 'D',  
                      '05': 'E', '06': 'H', '07': 'L', '08': 'M',  
                      '09': 'P', '10': 'R', '11': 'S', '12': 'T'}  
    return dizionarioMesi[parola]
```

# Sesso

- I due caratteri numerici indicativi del giorno di nascita e del sesso vengono determinati nel modo seguente:
  - Per i **soggetti maschili** il giorno di nascita figura **invariato**, con i numeri da uno a trentuno, facendo precedere dalla cifra zero i giorni del mese dall'uno al nove.
  - Per i **soggetti femminili** il giorno di nascita viene **aumentato di quaranta unità**, per cui esso figura con i numeri da quarantuno a settantuno.

```
def estraiCarGiorno(parola, carGenere):  
    if carGenere == 'F':  
        giorno = int(parola) + 40  
        print giorno  
        parola = str(giorno)  
    return parola
```



# Luogo

- I quattro caratteri alfanumerici indicativi del comune italiano o dello Stato estero di nascita, sono costituiti da un carattere alfabetico seguito da tre caratteri numerici, secondo la [codifica stabilita dall'Agenzia del Territorio e contenuta nel file CodiceComuniItalia.csv](#)

CODICI COMUNI D'ITALIA - COMUNI ATTUALI alla data del 6 febbraio  
Codice;Prov;Denominaz Italiana;Denominaz Estera;Cod Cat;Uff cat  
A154;VI;ALBETTONE;;D7AC;VI;VI;VI00;Vicenza;24002;;  
A155;CZ;ALBI;;T2AC;CZ;CZ;CZ00;Catanzaro;79002;;  
A157;TO;ALBIANO D'IVREA;;A1AF;TO;TO;TO40;Ivrea;1004;;  
A158;TN;ALBIANO;;E1AD;;;;;22002;;  
A159;MB;ALBIATE;;C1AI;MI;MI;MI20;Milano 2;108003;;  
A160;CS;ALBIDONA;;T3AH;CS;CS;CS00;Cosenza;78006;;  
A161;PD;ALBIGNASEGO;;D3AD;PD;PD;PD00;Padova;28003;;

[primo campo](#)

# Luogo

```
def estraiCarComune(parola):  
    dizComuni = {}  
    nomeFile = 'CFCodiceComuniItalia.csv'  
    fileComuni = open(nomeFile, 'r')  
    listaComuni = fileComuni.readlines()  
    fileComuni.close()  
    for linea in listaComuni:  
        frammenti = linea.split(';')  
        # print frammenti  
        dizComuni[frammenti[2]] = frammenti[0]  
    return dizComuni[parola.upper()]
```

# Carattere alfabetico di controllo

Il sedicesimo carattere ha funzione di controllo dell'esatta trascrizione dei primi quindici caratteri e viene determinato in questo modo:

- **ciascuno degli anzidetti** quindici caratteri, a seconda che occupi **posizione di ordine pari** o posizione di ordine **dispari**, viene **convertito** in un valore numerico, in base alle **tabelle di corrispondenza** successivamente riportate...

Per i sette caratteri con posizione di ordine

- **pari**: file **CFTabellaPari.tsv**
- **dispari**: file **CFTabellaDispari.tsv**

## CFTabellaPari.tsv

#Carattere	Valore	Carattere	Valore	Carattere	Valore
Valore	Carattere	Valore	Carattere	Valore	Carattere
0	0	9	9	I	8
1	1	A	0	J	9
2	2	B	1	K	10
3	3	C	2	L	11
4	4	D	3	M	12
5	5	E	4	N	13
6	6	F	5	O	14
7	7	G	6	P	15
8	8	H	7	Q	16
				R	17
				S	18
				T	19
				U	20
				V	21
				W	22
				X	23
				Y	24
				Z	25

## CFTabellaDispari.tsv

#Carattere	Valore	Carattere	Valore	Carattere	Valore
Valore	Carattere	Valore	Carattere	Valore	Carattere
0	1	9	21	I	19
1	0	A	1	J	21
2	5	B	0	K	2
3	7	C	5	L	4
4	9	D	7	M	18
5	13	E	9	N	20
6	15	F	13	O	11
7	17	G	15	P	3
8	19	H	17	Q	6
				R	8
				S	12
				T	14
				U	16
				V	10
				W	22
				X	25
				Y	24
				Z	23

# Controllo

```
def estraiCarControllo(parola):  
    file = open('CFTabellaDispari.tsv', 'r')  
    listaDispari = file.readlines()  
    file.close()  
    file = open('CFTabellaPari.tsv', 'r')  
    listaPari = file.readlines()  
    file.close()  
    del listaDispari[0]  
    del listaPari[0]  
    dizDispari = {}  
    for linea in listaDispari:  
        frammenti = linea.strip().split('\t')  
        for i in range(4):  
            dizDispari[frammenti[2 * i]] = frammenti[2 * i + 1]  
    # print dizDispari  
    dizPari = {}  
    for linea in listaPari:  
        frammenti = linea.strip().split('\t')  
        for i in range(4):  
            dizPari[frammenti[2 * i]] = frammenti[2 * i + 1]
```

# Carattere alfabetico di controllo

- I valori numerici così determinati vengono addizionati e la somma si divide per il numero 26.
- Il carattere di controllo si ottiene convertendo il resto di tale divisione nel carattere alfabetico ad esso corrispondente nel file `CFTabellaResto.tsv`

## CFTabellaResto.tsv

0	A
1	B
2	C
3	D
4	E
5	F
6	G
7	H
8	I
9	J
10	K
11	L
12	M
13	N
14	O
15	P
16	Q
17	R
18	S
19	T
20	U
21	V
22	W
23	X
24	Y
25	Z

# Controllo

```
file = open('CFTabellaResto.tsv', 'r')
listaResto = file.readlines()
file.close()
dizResto = {}
for linea in listaResto:
    frammenti = linea.strip().split('\t')
    dizResto[frammenti[0]] = frammenti[1]
# print dizResto
i = 0
somma = 0
for el in parola:
    if i % 2 == 1: # attenzione i è indice: 0 = primo
        somma += int(dizPari[el])
        print dizPari[el]
    else:
        somma += int(dizDispari[el])
        print dizDispari[el]
    i += 1
print 'somma = ', somma, 'resto = ', somma % 26
car = dizResto[str(somma % 26)]
return car
```



```
nome = raw_input('inserire il nome ')
cognome = raw_input('inserire il cognome ')
dataNascita = raw_input('inserire la data di nascita nel formato
    ggmmaaaa ')
genere = raw_input('inserire il genere M o F ')
comune = raw_input('inserire il comune di nascita ')

nome = nome.replace(' ', '')
print nome
cognome = cognome.replace(' ', '')
print cognome
giorno = dataNascita[:2]
print giorno
mese = dataNascita[2:4]
print mese
anno = dataNascita[6:8]
print anno, "\n"
carCognome = estraiCarCognome(cognome)
carNome = estraiCarNome(nome)
carMese = estraiCarMese(mese)
carGiorno = estraiCarGiorno(giorno, genere)
carComune = estraiCarComune(comune)
CF = carCognome + carNome + anno + carMese + carGiorno +
    carComune
carControllo = estraiCarControllo(CF)
CF += carControllo
print CF
```

## Dall'RNA alle proteine

I moderni strumenti di laboratorio permettono di estrarre le sequenze di nucleotidi (RNA) espresse in un certo individuo. Queste informazioni vengono salvate in file di formato standard FASTA. Sapendo che i nucleotidi vengono letti in triplette (codoni) e che ciascuna di queste triplette corrisponde ad un ammino-acido nella sequenza proteica, si chiede di scrivere un programma che abbia come input il file fasta `A06662-RNA.fasta` e la tabella di conversione `codonTable` e restituisca le possibili `sequenze proteiche`. Inoltre, per chiarezza, si sostituiscano gli ammino-acidi 'STOP' con il carattere '\*'

## A06662-RNA.fasta

[illegible]

# codonTable

GCU	A
GCC	A
GCA	A
GCG	A
CGU	R
CGC	R
CGA	R
...	
UUC	F
UAU	Y
UAC	Y
AUG	M
UGG	W
UAG	STOP
UGA	STOP
UAA	STOP

# fastaRNA2Prot.py

```
with open('codonTable', 'r') as codons:
    rna2cod = codons.readlines()
    codons.close()
# print rna2cod

dizRNA2Prot = {}
for line in rna2cod:
    tokens = line.strip().split('\t')
    dizRNA2Prot[tokens[0]] = tokens[1]
# print dizRNA2Prot

with open('A06662-RNA.fasta', 'r') as rna:
    rnaFile = rna.readlines()
    rna.close()
del rnaFile[0]
# print rnaFile

rnaString = ''
for line in rnaFile:
    rnaString += line.strip()
print rnaString
```

# fastaRNA2Prot.py

```
prots = []
for readFrame in range(3):
    prot = ''
    for i in range(readFrame, len(rnaString) - readFrame - 1,
3):
        codon = rnaString[i:i + 3]
        protFrag = dizRNA2Prot[codon]
        if protFrag == 'STOP':
            prot += '*'
        else:
            prot += protFrag
    prots.append(prot)
for prot in prots:
    print prot
```

## Cifratura per trasposizione ad inferriata

uenaacfarsaos ???

## Cifratura per trasposizione ad inferriata

u	e	n	a	a	c	f	a	r	s	a	o	s
---	---	---	---	---	---	---	---	---	---	---	---	---



## Cifratura per trasposizione ad inferriata

u	e	n	a	a	c	f	a	r	s	a	o	s
---	---	---	---	---	---	---	---	---	---	---	---	---

## Cifratura per trasposizione ad inferriata

u	e	n	a	a	c	f	a	r	s	a	o	s
---	---	---	---	---	---	---	---	---	---	---	---	---

unafras eacaso

## Cifratura per trasposizione ad inferriata: l'algoritmo

u	n	a	f	r	a	s	e	a	c	a	s	o
---	---	---	---	---	---	---	---	---	---	---	---	---

## Cifratura per trasposizione ad inferriata: l'algoritmo

u	n	a	f	r	a	s	e	a	c	a	s	o
---	---	---	---	---	---	---	---	---	---	---	---	---

si decide il taglio a metà frase

u	n	a	f	r	a	s	e	a	c	a	s	o
---	---	---	---	---	---	---	---	---	---	---	---	---

# Cifratura per trasposizione ad inferriata: l'algoritmo

u	n	a	f	r	a	s	e	a	c	a	s	o
---	---	---	---	---	---	---	---	---	---	---	---	---

si decide il taglio a metà frase

u	n	a	f	r	a	s	e	a	c	a	s	o
---	---	---	---	---	---	---	---	---	---	---	---	---

si decide allineano i 2 frammenti

u	n	a	f	r	a	s
e	a	c	a	s	o	

# Cifratura per trasposizione ad inferriata: l'algoritmo

u	n	a	f	r	a	s	e	a	c	a	s	o
---	---	---	---	---	---	---	---	---	---	---	---	---

si decide il taglio a metà frase

u	n	a	f	r	a	s	e	a	c	a	s	o
---	---	---	---	---	---	---	---	---	---	---	---	---

si decide allineano i 2 frammenti

u	n	a	f	r	a	s
e	a	c	a	s	o	

si scrive una lettera da un frammento di seguita ad una dell'altro

u	e	n	a	a	c	f	a	r	s	a	o	s
---	---	---	---	---	---	---	---	---	---	---	---	---

# Cifratura per trasposizione ad inferriata: l'algoritmo

u	n	a	f	r	a	s	e	a	c	a	s	o
---	---	---	---	---	---	---	---	---	---	---	---	---

si decide il taglio a metà frase

u	n	a	f	r	a	s	e	a	c	a	s	o
---	---	---	---	---	---	---	---	---	---	---	---	---

si decide allineano i 2 frammenti

u	n	a	f	r	a	s
e	a	c	a	s	o	

si scrive una lettera da un frammento di seguita ad una dell'altro

u	e	n	a	a	c	f	a	r	s	a	o	s
---	---	---	---	---	---	---	---	---	---	---	---	---

# Cifratura per trasposizione ad inferriata: l'algoritmo

u	n	a	f	r	a	s	e	a	c	a	s	o
---	---	---	---	---	---	---	---	---	---	---	---	---

si decide il taglio a metà frase

u	n	a	f	r	a	s	e	a	c	a	s	o
---	---	---	---	---	---	---	---	---	---	---	---	---

si decide allineano i 2 frammenti

u	n	a	f	r	a	s
e	a	c	a	s	o	

si scrive una lettera da un frammento di seguita ad una dell'altro

u	e	n	a	a	c	f	a	r	s	a	o	s
---	---	---	---	---	---	---	---	---	---	---	---	---



# Cifratura per trasposizione ad inferriata: l'algoritmo

u	n	a	f	r	a	s	e	a	c	a	s	o
---	---	---	---	---	---	---	---	---	---	---	---	---

si decide il taglio a metà frase

u	n	a	f	r	a	s	e	a	c	a	s	o
---	---	---	---	---	---	---	---	---	---	---	---	---

si decide allineano i 2 frammenti

u	n	a	f	r	a	s
e	a	c	a	s	o	

si scrive una lettera da un frammento di seguita ad una dell'altro

u	e	n	a	a	c	f	a	r	s	a	o	s
---	---	---	---	---	---	---	---	---	---	---	---	---

```

messaggio = raw_input('inserisci il messaggio ')

# CODIFICA
messaggioT = messaggio.replace(' ', '')
print messaggioT

iSpezza = (len(messaggioT) + 1) / 2
print 'len = ', len(messaggioT), 'iSpezza', iSpezza
stringa1 = messaggioT[:iSpezza]
stringa2 = messaggioT[iSpezza:]
print stringa1, len(stringa1)
print stringa2, len(stringa2)
lMessaggioCifrato = list(messaggioT)

for i in xrange(iSpezza):
    indice = i * 2
    print indice
    lMessaggioCifrato[indice] = stringa1[i]
    print ' ', indice + 1
    if i < len(stringa2):
        lMessaggioCifrato[indice + 1] = stringa2[i]

messaggioCifrato = ''.join(lMessaggioCifrato)
print messaggioCifrato

```

```
# DECODIFICA
iSpezza = (len(messaggioCifrato) + 1) / 2
frammento1 = ''
frammento2 = ''
for i in xrange(len(messaggioCifrato)):
    if i % 2 == 0:
        frammento1 += messaggioCifrato[i]
    else:
        frammento2 += messaggioCifrato[i]

print frammento1, frammento2

messaggioDecifrato = frammento1 + frammento2
print messaggioDecifrato
```