

1 Cate ceva despre reprezentarea caracterelor în calculator

Codul ASCII este o tabelă universal recunoscută unde fiecărui caracter (literă mare, literă mică, cifră sau caracter special) îi corespunde câte un număr.

Acesta poate fi scris în baza 16 (Hex) sau în baza 10 (Dec)

De exemplu:

literii A îi corespunde codul 65.

codul 97 corespunde literei a.

Cauta pe Google codul ASCII, images!

2 Cum am codifica manual un text, prin decalarea literelor din alfabet cu o poziție? Dar cu două poziții? A devine C, B devine D, s.a.m.d

3 Să scriem programul de criptare a unui text

Ne vom folosi de două funcții importante din Python.

Funcția ord() returnează codul ASCII al unui caracter.

Funcția chr() returnează caracterul care corespunde unui anumit număr.

Dec	Hex	Char	Dec	Hex	Char
64	40	@	96	60	`
65	41	A	97	61	a
66	42	B	98	62	b
67	43	C	99	63	c
68	44	D	100	64	d
69	45	E	101	65	e
70	46	F	102	66	f
71	47	G	103	67	g
72	48	H	104	68	h
73	49	I	105	69	i
74	4A	J	106	6A	j
75	4B	K	107	6B	k
76	4C	L	108	6C	l
77	4D	M	109	6D	m
78	4E	N	110	6E	n
79	4F	O	111	6F	o
80	50	P	112	70	p
81	51	Q	113	71	q
82	52	R	114	72	r
83	53	S	115	73	s
84	54	T	116	74	t
85	55	U	117	75	u
86	56	V	118	76	v
87	57	W	119	77	w
88	58	X	120	78	x
89	59	Y	121	79	y
90	5A	Z	122	7A	z
91	5B	[123	7B	{

KEY= 2

tx_out = input("Introdu textul tau:")

tx_out=""

for litera in tx_in:

num = ord(litera)

num = num + KEY

litera_noua = chr(num)

tx_out=tx_out + litera_noua

print("Text original: ", tx_in)

print("Text criptat: ", tx_out)

Am citit textul de la tastatură

Am inițializat o variabilă string goală.

Într-o structură FOR vom parcurge fiecare caracter din text.

Aflăm care e codul său ASCII, îl modificăm cu valoarea KEY, adică cu 2, și apoi transformăm înapoi în caracter. Lipim acest caracter la șirul gol inițiat la început.

La părăsirea buclei FOR vom avea deja noul text complet, construit literă cu literă.

Afișăm ambele texte, unul sub altul.

65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[\
C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B		
97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124
a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	{	
c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b		

4 Dezvoltarea no.1 – numai literele

Dorim să codificăm doar literele, caracterele speciale să le lășăm așa cum sunt. Literele mari să fie convertite tot în litere mari, literele mici să fie convertite tot în litere mici.

```
for litera in tx_in:
    litera_noua = lit
    if litera.isalpha():
        num = ord(litera)
        num = num + KEY
    tx_out=tx_out + litera_noua
```

Am folosit metoda `isalpha()` care testează dacă un string conține numai litere. Dacă rezultatul e `False`, nu mai Codificăm nimic.

5 Dezvoltarea no.2 - codificare circulară

Să avem grijă ca această codificare să fie circulară, adică litera Z să se transforme în litera A (dacă KEY este 1) sau în litera B (dacă KEY este 2), s.a.m.d. Pentru literele mici, la fel. Trebuie doar să analizăm cu atenție tabelul din pagina precedentă și o să știm ce avem de făcut

```
for litera in tx_in:
    litera_noua = lit
    if litera.isalpha():
        num = ord(litera)
        num = num + KEY
        if litera.isupper() and num > ord("Z"):
            num = num -26
        if litera.islower() and num > ord("z"):
            num = num -26
        litera_noua = chr(num)
    tx_out=tx_out + litera_noua
```

Dacă prin transformare obținem un cod ASCII mai mare decât 90, adică **codul literei Z**, atunci trebuie să sărim înapoi la începutul alfabetului, adică scădem numărul 26. La fel pentru litere mici, dacă prin transformare obținem un cod ASCII mai mare decât 122, adică **codul literei z**, atunci trebuie să sărim la începutul alfabetului de litere mici, adică scădem numărul 26.

6 Dezvoltarea no.3 - Introducem totul într-o funcție

```
KEY= 2
def crypto_in(mess):
    tx_in = mess
    tx_out=""
    for litera in mess:
        litera_noua = litera
        if lit.isalpha():
            num = ord(litera)
            num = num + KEY
            if lit.isupper() and num > ord("Z"):
                num = num -26
            if lit.islower() and num > ord("z"):
                num = num -26
            litera_noua = chr(num)
        tx_out=tx_out + litera_noua
    return tx_out
```

```
mesaj = input("Introdu textul tau:")
rezult = crypto_in(mesaj)
```

Observa cum am definit și cum am apelat funcția `crypto_in`. Funcția nu se va executa decât atunci când o apelăm, adică în ultimul rând. Acolo se atribuie variabilei **rezult** rezultatul prelucrării din interiorul funcției (adică ce se trimite prin `return`)



Julius Caesar