Algorítimo da Cifra de César

Roberto Sadao Yokoyama

UTFPR-CP

Agosto, 2016

Prática de hoje:[1]

[1] Baseado no código: https://inventwithpython.com/chapter14.html

Usando números para letras

- Tabela ASCII

32	(space)	48	0	64	@	80	P	96	`	112	p
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	'	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	|
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o		

Funções *ord* e *chr*

```
>>> chr(65)
'A'
>>> ord('A')
65
>>> chr(65+8)
'I'
>>> chr(52)
'4'
>>> chr(ord('F'))
'F'
>>> ord(chr(68))
68
```

Definindo a função encrypt

```
def encrypt(k,plaintext):
    cipher=''
    for c in plaintext:
        c=(ord(c)+k)
        cipher+=chr(c)
    return cipher
```


Definindo a função decrypt

```
def decrypt(k,cipher):
    plaintext=''
    for c in cipher:
        c=(ord(c)-k)
        plaintext+=chr(c)
    return plaintext
```


Testando com 'hello word!'

```
plaintext='hello word'
k=3
e= encrypt(k,plaintext)
print e
print decrypt(k,e)
```


Implemente

1) Implemente/Modifique o código para que as funções de encriptar e decriptar fiquem de acordo com o algoritmo da cifra de César. Por exemplo: para $k = 3$, a letra **z** (cód. 122) deve ser substituída pela letra **c** (cód. 99) e não pelo caractere } (cód. 125). Além disso, limite o tamanho da chave secreta para no máximo $k = 26$ e adeque a correspondência da chaves k.
2) Crie uma função *brute_force* que testa todas as chaves, k, possíveis. Mostre as saídas para cada chave de maneira a poder identificar a chave secreta utilizada.