Funciones PYTHON 

Funciones interesantes

str.**maketrans** y **translate**

```
testword = str.maketrans('aei' + 'aei'.upper() ,"AEI" + "AEI".lower())
```

*Lo que hace **maketrans** es hacer el translado de palabras a su CODIGO ASCII y hacer una especie de **diccionario** con la primer letra del primer parametro y la primer letra del segundo parametro, y así con todos los restantes.*

```
{97: 65, 101: 69, 105: 73, 65: 97, 69: 101, 73: 105}
a    A    e    E   i   I    A   a   E   e   I    i
```

97 es 'a' y 65 es 'A'

Esto nos va a servir para hacer la traduccion de nuestra palabra con **translate()**

```
new_word = 'cACa'.translate(testword)
print(new_word)
```

*tenemos la palabra "cACa" y vamos a usar **translate** y le vamos a pasar como parametro al diccionario que hemos creado: testword*

Lo que estamos por hacer es cuando tenemos 'A', nos pasará a: 'a'. y viceversa. Tenemos finalmente este resultado:

```
caCA
```

---

for

Formas de usarlo:

```
numeros = [1,2,3,4,5]

for num in numeros:
    if num % 2 == 0:
        print(f"{num} es par")
```

```
even_numbers = [num for num in range(21) if num % 2 == 0]
print(even_numbers)
```

Funciones Lambda

Se las conoce como ***funciones anonimas*** (porque no llevan nombre) y son una alternativa a las funciones creadas con ***def***. Se las usan normalmente dentro de funciones complejas como ***filter()*** o ***map()***.

```
numbers = [1, 2, 3, 4, 5]

even_numbers = list(filter(lambda x: x % 2 == 0, numbers))
print(even_numbers)  # [2, 4]

squared_numbers = list(map(lambda x: x ** 2, numbers))
print(squared_numbers)
```

output

```
[2, 4]
[1, 4, 9, 16, 25]
```

Listas

append() y **insert()**

```
languages = ['Spanish', 'English', 'Japanese']
languages.append('Chinese')

print(languages)

languages.insert(0,'Cantonese')

print(languages)
```

output

```
#Append (final)
['Spanish', 'English', 'Japanese', 'Chinese']
#Insert index 0
['Cantonese', 'Spanish', 'English', 'Japanese', 'Chinese']
```

---

enumerate

```
languages = ['Spanish', 'English', 'Japanese']
enumerate_languages = list(enumerate(languages), start = 1)
```

output

```
[(1, 'Spanish'), (2, 'English'), (3, 'Japanese')]
```

*Como vemos, **enumerate()** nos crea una lista de tuplas con numero de orden y el valor de item de la lista.*

Otro uso:

```
for index, lang in enumerate(languages, start = 1):
    print(index,lang)
```

Zip

```
a = ("John", "Charles", "Mike")
b = ("Jenny", "Christy", "Monica", "Qoqo")

x = list(zip(a, b))

print(x)
```

*Como vemos, **zip()** es similar a **enumerate()** pero podemos tenemos que pasarle dos parámetros para que los una.*

output

```
[('John', 'Jenny'), ('Charles', 'Christy'), ('Mike', 'Monica')]
```

En este caso, vemos que "Qoqo" sobra y no lo imprime porque no puede unirlo con otro valor del primer parámetro (a)