Clase_Iteradores

June 8, 2023

1 Seminario de Lenguajes - Python

1.1 Iteradores

2 ¿Qué observan en los siguientes códigos?

[1]:
```
cadena = "Seminario de Python"
for caracter in cadena:
    print(caracter, end="-")
print("", end="\n")

print("\n")
lista = ['esto', 'es', 'una', 'lista']
for palabra in lista:
    print(palabra)
```

```
S-e-m-i-n-a-r-i-o- -d-e- -P-y-t-h-o-n-


esto
es
una
lista
```

[ ]:
```
import csv

with open("Pokemon.csv", encoding='utf-8') as data_set:
    reader = csv.reader(data_set, delimiter=',')
    for item in reader:
        print(item)
```

3 Todas son secuencias iterables

3.1 ¿Qué significa?

- Todas pueden ser recorridas por la estructura: **for** var **in** secuencia.
- Todas implementan un método especial denominado **___iter___**.

– **___iter___** devuelve un iterador capaz de recorrer la secuencia.

Un **iterador** es un objeto que permite recorrer **uno a uno** los elementos de una estructura de datos para poder operar con ellos.

4 Iteradores

- Un objeto iterable tiene que implementar un método **___next___** que debe devolver los elementos, **de a uno por vez**, comenzando por el primero.
- Y al llegar al final de la estructura, debe levantar una excepción de tipo **StopIteration**.

5 Los siguientes códigos son equivalentes:

[3]:
```
lista = ['uno', 'dos', 'tres']
for palabra in lista:
    print(palabra)
```

```
uno
dos
tres
```

[4]:
```
iterador = iter(lista)
while True:
    try:
        palabra = next(iterador) # o iterador.__next__()
    except StopIteration:
        break
    print(palabra)
```

```
uno
dos
tres
```

- La función **iter** retorna un objeto iterador.

6 Veamos este ejemplo:

[5]:
```
class CadenaInvertida:
    def __init__(self, cadena):
        self._cadena = cadena
        self._posicion = len(cadena)

    def __iter__(self):
        return(self)

    def __next__(self):
        if self._posicion == 0:
```

```
            raise(StopIteration)
        self._posicion = self._posicion - 1
        return(self._cadena[self._posicion])
```

[6]:
```
cadena_invertida = CadenaInvertida('Ya estamos al final de la cursada!!')

for caracter in cadena_invertida:
    print(caracter, end=' ')
```

```
! ! a d a s r u c   a l   e d   l a n i f   l a   s o m a t s e   a Y
```

- ¿Qué creen que imprime?

7 Desafío

Implementar la clase **CadenaCodificada**, que dada una cadena de caracteres, me permita trabajar con la misma en forma codificada, según la codifcación Cesar (vista en las primeras clases). Podemos recorrer con un **for** un objeto de clase **CadenaCodificada**, los cual permite acceder uno a uno a los caracteres codificados de la misma.

Nota: implementar este objeto como un objeto iterable.

[8]:
```
from functools import reduce

class CadenaCodificada:
    ....
```

7.1 Ejemplo de uso de una CadenaCodificada

[9]:
```
mi_cadena = CadenaCodificada("Hola")

for caracter in mi_cadena:
    print(caracter, end=" ")
```

```
I p m b
```

[10]:
```
print(mi_cadena)
```

```
Ipmb
```