Programació amb Python

Unió Europea
Fons social europeu

Tema 6- Colecciones

6.1 Tuplas

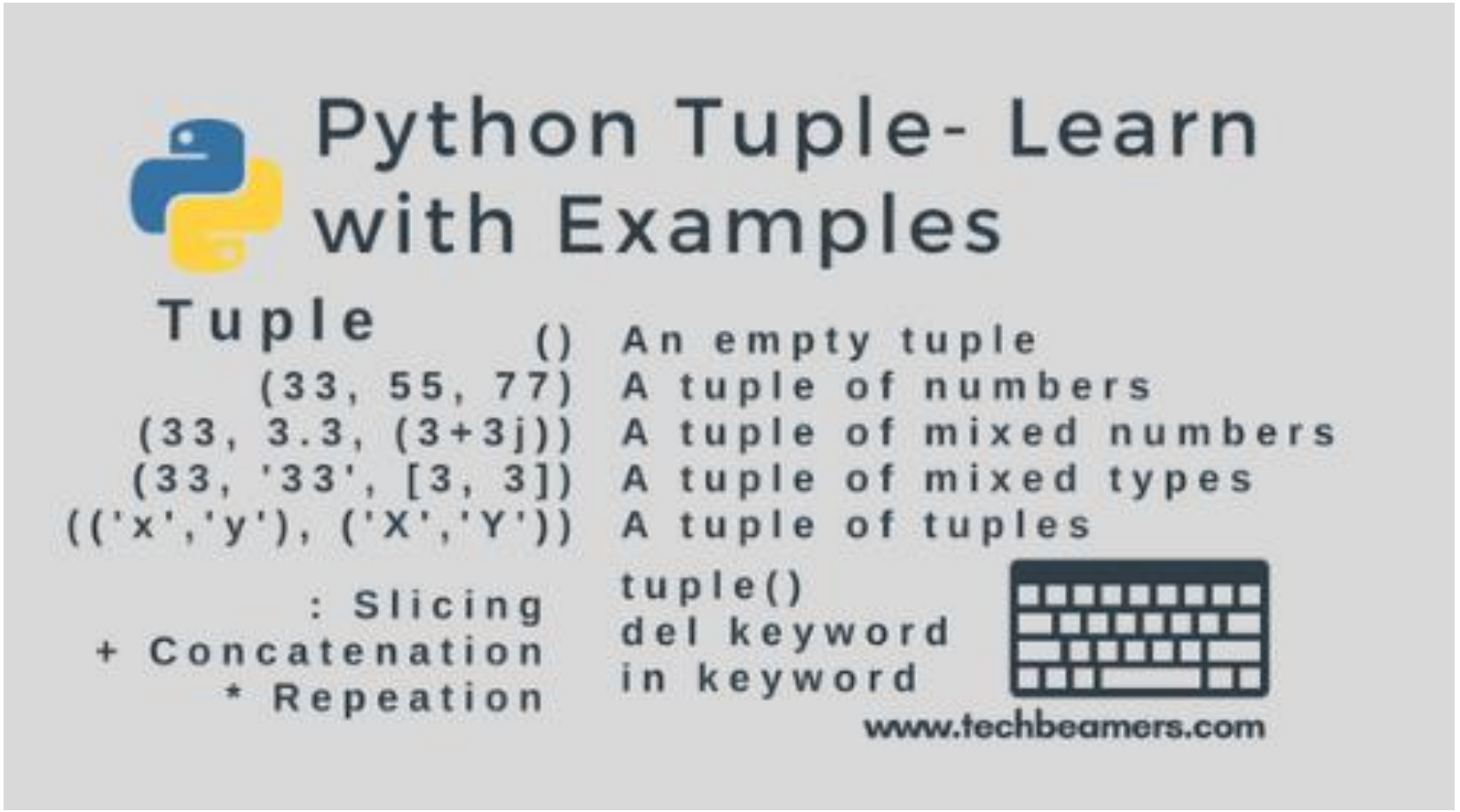
6.2. Diccionarios

- Tuplas
- Conjuntos
- Diccionarios

- Pilas
- Colas

simulables con listas

6.1. Tuplas


Python Tuple- Learn with Examples
Tuple
() An empty tuple
(33, 55, 77) A tuple of numbers
(33, 3.3, (3+3j)) A tuple of mixed numbers
(33, '33', [3, 3]) A tuple of mixed types
(('x','y'), ('X','Y')) A tuple of tuples
: Slicing
+ Concatenation
* Repeation
tuple()
del keyword
in keyword
www.techbeamers.com

Qué son las tuplas ?

Se parecen a las listas pero son inmutables.
En lugar de usar corchetes se usan paréntesis.

```
tupla = (100,"Hola",)
```

Son más rápidas que las listas. Adecuadas para variables que no cambian :

nomMes (“gener”, “febrer”, “març”)

lienzo_XY_ini (0,300)

lienzo_XY_fin (600,800)

Ubicaciones ((20.0213N, 34.67E), (206.87N, 604.67O))

Tuplas con índices y slicing

```
In [1]: tupla = (100,"Hola",[1,2,3,4],-50)

In [2]: tupla
Out[2]: (100, 'Hola', [1, 2, 3, 4], -50)

In [3]: tupla[0]
Out[3]: 100

In [4]: tupla[-1]
Out[4]: -50

In [6]: tupla[2:]
Out[6]: ([1, 2, 3, 4], -50)

In [8]: tupla[2][-1]
Out[8]: 4

In [9]: tupla[0] = 50
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-9-b45433b4cee9> in <module>()
----> 1 tupla[0] = 50

TypeError: 'tuple' object does not support item assignment
```

```
In [10]: len(tupla)
Out[10]: 4

In [13]: len(tupla[2])
Out[13]: 3

In [15]: # .index para buscar un elemento y saber su posición
         tupla.index(100)
Out[15]: 0

In [16]: tupla
Out[16]: (100, 'Hŏla', [1, 2, 3], -50)

In [17]: tupla.index('Hola')
Out[17]: 1

In [18]: tupla.index('Otro')
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-18-640d616163a2> in <module>()
----> 1 tupla.index('Otro')

ValueError: tuple.index(x): x not in tuple
```

len

.index

```
(100, 'Hola', [1, 2, 3], -50)
```

```
In [19]: tupla.count(100)
```

```
Out[19]: 1
```

```
In [20]: tupla.count('Algo')
```

```
Out[20]: 0
```

.count

```
In [21]: tupla = (100,100,100,50,10)
```

```
In [22]: tupla.count(100)
```

```
Out[22]: 3
```

```
In [23]: tupla.append(10)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-23-758d195ec9d7> in <module>()
----> 1 tupla.append(10)

AttributeError: 'tuple' object has no attribute 'append'
```

Se pueden crear tuplas n-dimensionales, pero no se pueden variar.

```
t1 = "08", "Barcelona", "Catalunya"
t2 = "27", "Girona", "Catalunya"
todas = t1,t2
print (todas)
```

```
(('08', 'Barcelona', 'Catalunya'), ('27', 'Girona', 'Catalunya'))
```

Podemos crear dinámicamente una lista

```
t = [i for i in range(5)]
print (t)
```

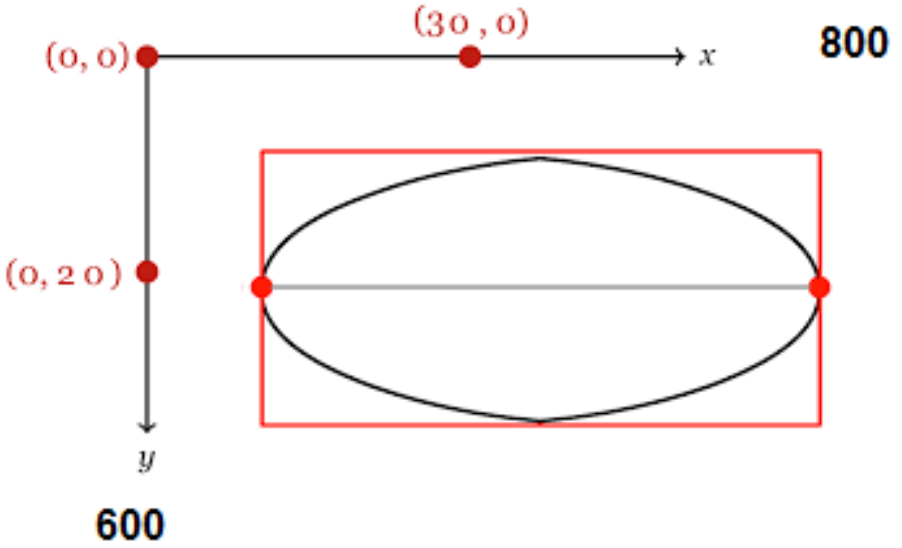
Esto funciona con listas

```
[0, 1, 2, 3, 4]
```

Comprueba si funciona con tuplas :

```
t = (i for i in range(5))
print (t)
```

Prepara un cuaderno jupyter con estas prácticas



Practica P01: Elipse

Estudia el siguiente programa, cuántas tuplas ves ?

```
from PIL import Image, ImageDraw

img = Image.new('RGB', (600, 400), 'White')
dib = ImageDraw.Draw(img)
#------------- dibuja una elipse y la muestra en el lienzo
dib.ellipse((150, 50, 449, 349), 'Crimson')  # (x0, y0, x1, y1)
img.show()
#------------- salva la imagen  y su miniatura
img.save('img\japon.jpg', "JPEG")
size = (128, 128)
img.thumbnail(size)
img.save('img\japon-mini.jpg', "JPEG")
```

Realiza los cambios

1. Medida de la figura original a 500 x 500 color 'darkgoldenrod'
2. Dibujar una elipse en la misma posición del ejercicio anterior pero color 'Black'
3. Dibujar una nueva elipse des de (50,70) a (150, 170) en color 'lavender'
4. Salva los ficheros como ikusa.jpg e ikusa-mini.jpg

Práctica P02-Tuplas

1- Tenemos una tupla con datos de una persona :

t= 89766, "Alicia", "Hacker”

Muestra los datos y pide año de registro.

Puedes añadir el año a la cola de esta tupla ?

Como puedes conseguir una tupla con los datos de la persona y el año de registro ?

2- Programa la instrucción para mostrar los elementos 2ndo y 3ero

3- Avanzado : Crea una tupla tipo “personas = ((codigo, nombre, apellidos, anyo_reg))”

con los datos de Alicia más los tuyos.

Nota : Usa como “código” -> tu “dni” sin la letra. Muestra la lista de personas del ejercicio anterior

4- Crea una matriz tal que :

- las columnas sean las temperaturas medias de los últimos 5 días en una ubicación dada :

```
"Barcelona", 27, 28, 26, 24, 22
```

- las filas sean 4 ubicaciones geográficas distintas

Pide por pantalla una ubicación y muestra las temperaturas de los 5 días y la media de los 5 valores.

Si la ubicación es = "*" entonces muestra todas las ubicaciones.

5- Avanzado : perfecciona el programa para que al empezar muestre la lista de ubicaciones disponibles y al comparar las ubicaciones existentes con la introducida se insensible a mayúsculas y minúsculas. `Ver string.upper()`

Práctica P03 – Métodos de las tuplas - Avanzado

```
[("Manuel Juarez", 19823451, "Liverpool"),
("Silvana Paredes", 22709128, "Buenos Aires"),
("Rosa Ortiz", 15123978, "Glasgow"),
("Luciana Hernandez", 38981374, "Lisboa")]
```

```
[("Buenos Aires","Argentina"),
("Glasgow","Escocia"), ("Lisboa", "Portugal"),
("Liverpool","Inglaterra"), ("Madrid","España")]
```

1-Agregar pasajero : Agregar pasajeros a la lista de viajeros.
2-Agregar Ciudad : Agregar ciudades a la lista de ciudades.

3-Destino - Pasajero: Dado el DNI de un pasajero, ver a qué país y ciudad viaja.

4-Pasajeros x Ciudad : Dada una ciudad, mostrar la cantidad de pasajeros que viajan a esa ciudad.

5-Pasajeros x País : Dado un país, mostrar cuántos pasajeros viajan a ese país.

0-Salir del programa.

Soluciones

#P01 – Elipse----------

```
from PIL import Image, ImageDraw

img = Image.new('RGB', (500, 500), 'darkgoldenrod')
dib = ImageDraw.Draw(img)

#-------------- dibuja una elipse y la muestra en el lienzo
dib.ellipse((150, 50, 449, 349), 'Black')    # (x0,y0, x1, y1)
dib.ellipse((70, 50, 170, 150), 'lavender')  # (x0,y0, x1, y1)
img.show()
#----------------- salva la imagen  y su miniatura
img.save('img\ikusa.jpg', "JPEG")
size = (128, 128)
img.thumbnail(size)
img.save('img\ikusa-mini.jpg', "JPEG")
```

Visita estos enlaces,
https://pillow.readthedocs.io/en/stable/handbook/tutorial.html#create-jpeg-thumbnails

Como crear todas las miniaturas de una lista de imágenes pasada como argumento.

https://www.codegrepper.com/code-examples/python/plotly+colours

Soluciones

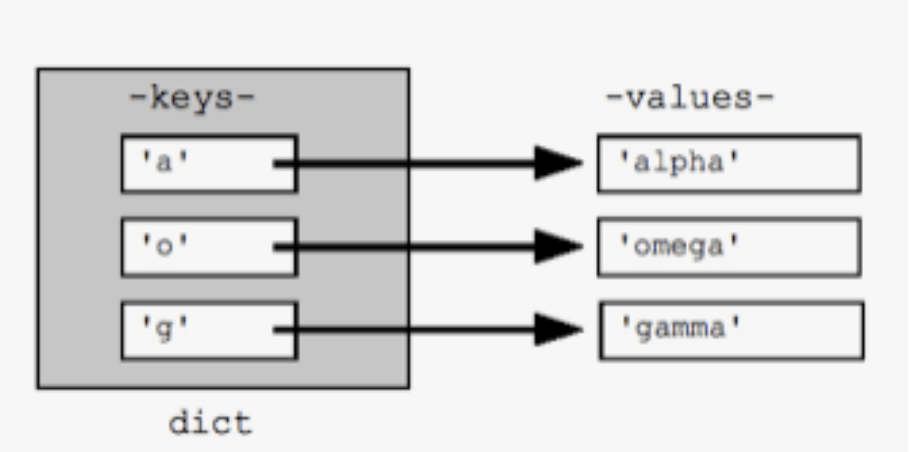
```
#P02 - Tuplas----------------------------------
#1.-
t= (89766, "Alicia", "Hacker")
any_reg = int(input("Año registro : "))
ta = t, any_reg
print (ta)

#P02----------------
#2.-
print(ta[0][1], ta[0][2])

#P02 - Tuplas----------------------------------
#3.-
p1 = (89766, "Alicia", "Hacker", 2010 )
p2 = (46227578, "Marc", " Martín Rus ", 2008)

persones = p1, p2
for p in persones :
   print (p)
```

Soluciones

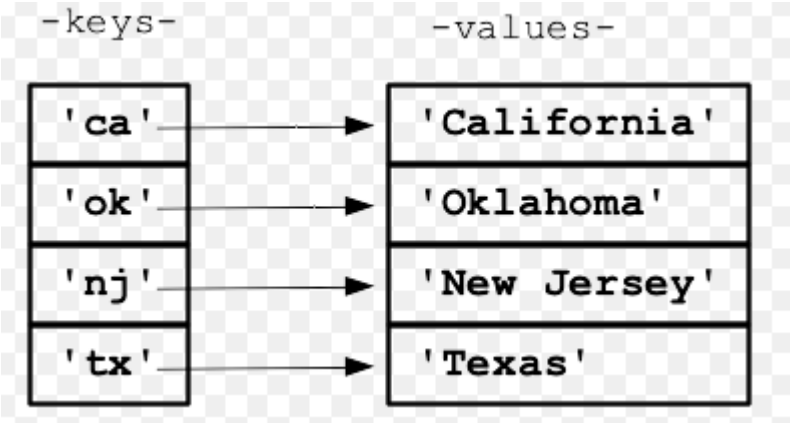
```
#P02 Exercici 4 ------------
u1 = "Barcelona", 27, 28, 26, 24, 22
u2 = "Tarragona", 29, 30, 29, 30, 28
u3 = "Lleida", 22, 22, 21, 20, 21
u4 = "Girona", 26, 25, 24, 22, 20

ubicacions = (u1,u2,u3,u4)
ubicacio = input("Ubicació : ")

for t in ubicacions :
    if t.count(ubicacio ) > 0
          or ubicacio =="*" :
       print (t)
```

```
#P02 Exercici 5 ------------
u1 = "Barcelona", 27, 28, 26, 24, 22
u2 = "Tarragona", 29, 30, 29, 30, 28
u3 = "Lleida", 22, 22, 21, 20, 21
u4 = "Girona", 26, 25, 24, 22, 20

ubicacions = (u1,u2,u3,u4)

print("Ubicacions disponibles")
print("---------------------")
for t in ubicacions :
    print (t[0])

ubicacio = input("Ubicació a consultar : ")
ubicacio = ubicacio .upper()
for t in ubicacions :
    if t[0].upper() == ubicacio or ubicacio =="*" :
       print (t)
```

6.2. Diccionarios




```
{"Name": "John",
 "Age": 21,
 "Id": 28}
```




```
{
 'name' : ['Joe', 'Nat', 'Harry'],
 'age'  : [20, 21, 19],
 'marks': [85.1, 77.8, 91.54]
}
```

	name	age	marks
0	Joe	20	85.1
1	Nat	21	77.8
2	Harry	19	91.54

Que son los diccionarios ?

Un diccionario es una colección no ordenada de objetos, que se componen de una clave y un valor.

- Las claves suelen ser números enteros o cadenas, aunque cualquier otro objeto inmutable puede actuar como una clave.
- Los valores, por el contrario, pueden ser de cualquier tipo, incluso otros diccionarios.

```
d = {"Python": 1991, "C": 1972, "Java": 1996}
```

```
d["Python"]
1991
```

El diccionario se define entre llaves cada elemento se define

clave : valor y separados por coma.

Usando la clave se obtiene el valor.

```
In [1]: vacio = {}

In [2]: vacio
Out[2]: {}

In [3]: type(vacio)
Out[3]: dict

In [4]: colores = {'amarillo':'yellow','azul':'blue', 'verde':'green'}

In [5]: colores
Out[5]: {'amarillo': 'yellow', 'azul': 'blue', 'verde': 'green'}

In [6]: colores['azul']
Out[6]: 'blue'

In [7]: colores['amarillo']
Out[7]: 'yellow'
```

```
colores['amarillo'] = 'white'
```

```
colores
```

```
{'amarillo': 'white', 'azul': 'blue', 'verde': 'green'}
```

```
numeros = {10:'diez',20:'veinte'}
```

```
numeros[10]
```

```
'diez'
```

```
In [14]: edades = {'Hector':27,'Juan':45,'Maria':34}

In [15]: edades
Out[15]: {'Hector': 27, 'Juan': 45, 'Maria': 34}

In [16]: edades['Hector']+=1

In [17]: edades
Out[17]: {'Hector': 28, 'Juan': 45, 'Maria': 34}

In [18]: edades['Juan'] + edades['Maria']
Out[18]: 79
```

```
In [19]: for edad in edades:
             print(edad)

Maria
Hector
Juan
```

```
In [20]: for clave in edades:
             print(edades[clave])

34
28
45
```

In [21]:
```
for clave in edades:
    print(clave,edades[clave])
```

```
Maria 34
Hector 28
Juan 45
```

In [22]:
```
for clave,valor in edades.items():
    print(clave,valor)
```

```
Maria 34
Hector 28
Juan 45
```

In [23]:
```
for c,v in edades.items():
    print(c,v)
```

```
Maria 34
Hector 28
Juan 45
```



```
personajes = []

gandalf = {'Nombre':'Gandalf','Clase':'Mago','Raza':'Humano'}
legolas = {'Nombre':'Legolas','Clase':'Arquero','Raza':'Elfo'}
gimli = {'Nombre':'Gimli','Clase':'Guerrero','Raza':'Enano'}

personajes.append(gandalf)
personajes.append(legolas)
personajes.append(gimli)

print(personajes)

[{'Clase': 'Mago', 'Nombre': 'Gandalf', 'Raza': 'Humano'},
 {'Clase': 'Arquero', 'Nombre': 'Legolas', 'Raza': 'Elfo'},
 {'Clase': 'Guerrero', 'Nombre': 'Gimli', 'Raza': 'Enano'}]
```

```
In [24]: personajes = []

In [25]: p = {'Nombre':'Gandalf','Clase':'Mago','Raza':'Humano'}

In [26]: personajes.append(p)

In [27]: personajes
Out[27]: [{'Clase': 'Mago', 'Nombre': 'Gandalf', 'Raza': 'Humano'}]

In [28]: p = {'Nombre':'Legolas','Clase':'Arquero','Raza':'Elfo'}

In [29]: personajes.append(p)

In [30]: p = {'Nombre':'Gimli','Clase':'Guerrero','Raza':'Enano'}

In [31]: personajes.append(p)

In [32]: personajes
Out[32]: [{'Clase': 'Mago', 'Nombre': 'Gandalf', 'Raza': 'Humano'},
          {'Clase': 'Arquero', 'Nombre': 'Legolas', 'Raza': 'Elfo'},
          {'Clase': 'Guerrero', 'Nombre': 'Gimli', 'Raza': 'Enano'}]
```

In [33]:
```
for p in personajes:
    print(p['Nombre'],p['Clase'],p['Raza'])
```

```
Gandalf Mago Humano
Legolas Arquero Elfo
Gimli Guerrero Enano
```

Práctica P04 – Personajes

Durante el desarrollo de un pequeño videojuego se te encarga configurar y balancear cada clase de personaje jugable. Partiendo que la estadística base es 2, debes cumplir las siguientes condiciones:

Caballero : doble vida y defensa que el guerrero
Guerrero : doble ataque y alcance que un caballero
Arquero = Misma vida y ataque que un guerrero,
pero la mitad de su defensa y el doble de su alcance

Muestra las propiedades de los 3 personajes.

caballero = {'vida':2 , 'ataque':2, 'defensa':2, 'alcance':2}
guerrero = {'vida':2 , 'ataque':2, 'defensa':2, 'alcance':2}
arquero = {'vida':2 , 'ataque':2, 'defensa':2, 'alcance':2}

Avanzado:

- Crea un programa que pida un personaje , una atributo y su valor numérico.

-Calcula los valores de los atributos del resto de los personajes usando los criterios mencionados en el ejercicio anterior

- Muestra los atributos de todos los personajes

EJEMPLO

```
Introduzca un personaje  (c=caballero, g=guerrero, a=arquero) :
Introduzca un atributo ( v=vida, at= ataque, d=defensa, al=alcance) :
Valor (0-99) :
```

Práctica P05 – Diccionaris

Estudiem aquest exemple :

```
import random
aleatoris = ["dilluns","dimarts","dimecres", "dijous", "divendres",
             "dissabte", "diumenge"]
random.shuffle(aleatoris)

print (aleatoris)


dic_fr = {"dilluns": "lundi", "dimarts":"mardi","dimecres":"mercredi",
          "dijous":"jeudi", "divendres":"vendredi",
          "dissabte":"samedi", "diumenge":"dimanche"}

for d in aleatoris :
    print (f" el dia {d:10} traduït és {dic_fr[d]} ")
```

Prova si funciona bé.

I ara continua amb la pràctica següent.

Práctica P05.1 – Diccionarios Idiomas

- Crea 3 diccionarios con los días de la semana en catalán/castellano, inglés y francés. Decide cual es el idioma clave y usa el mismo en todos los diccionarios.

- Pide por pantalla el idioma (0= salir).

- Muestra aleatoriamente los dias de la semana en el idioma solicitado

Métodos de los diccionarios

Tabla de contenidos


```
colores = { "amarillo":"yellow", "azul":"blue", "verde":"green" }
```

get()

Busca un elemento a partir de su clave y si no lo encuentra devuelve un valor por defecto:

Código Resultado

```
colores.get('negro','no se encuentra')
```

Practica P06 – Métodos de los diccionarios

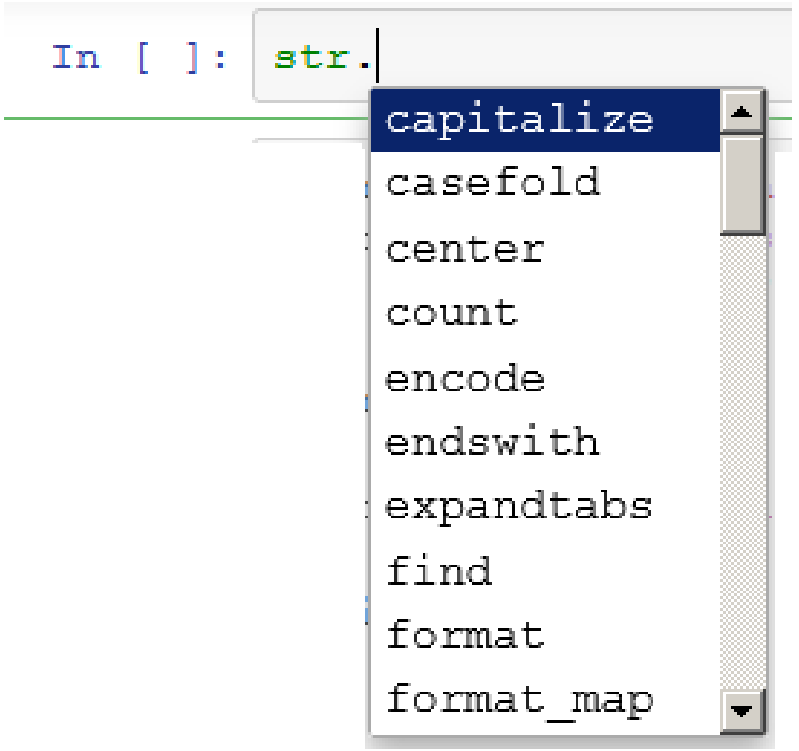
- diccionario
- fruta número de kilos
- importe
-

Fruta	Precio
Plátano	1.35
Manzana	0.80
Pera	0.85
Naranja	0.70

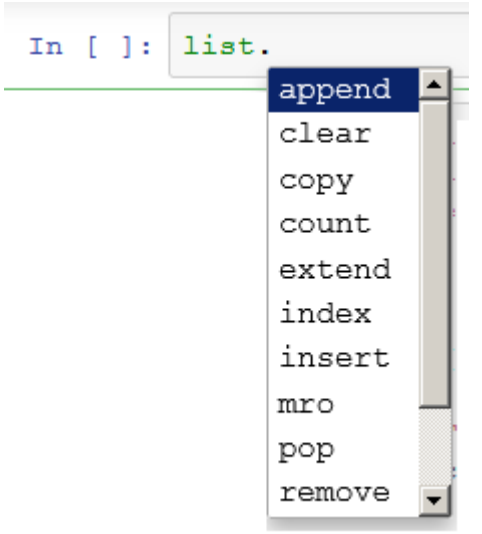
https://aprendeconalf.es/python/ejercicios/diccionarios.html

Repaso Cadenas y Listas

Métodos de las cadenas



Métodos de las listas



Repaso: Métodos en las cadenas

https://docs.hektorprofe.net/python/metodos-de-las-colecciones/metodos-de-las-cadenas/

<class 'str'>

upper() Devuelve la cadena con todos sus caracteres a mayúscula
lower() Devuelve la cadena con todos sus caracteres a minúscula

find() Devuelve el índice en el que aparece la subcadena (-1 si no aparece)

```python
cadena = "Hola Mundo"
print("islower ", cadena.islower())
print("isupper ", cadena.isupper())
print("isspace ", cadena.isspace())
print("istitle ", cadena.istitle())
```

```
islower  False
isupper  False
isspace  False
istitle  True
```

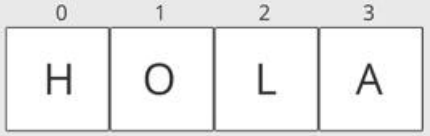
split() Separa la cadena en subcadenas a partir de sus espacios y devuelve una lista

replace() Reemplaza una subcadena de una cadena por otra y la devuelve

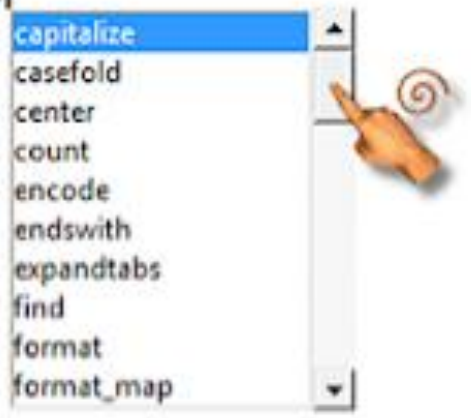
```
cadena = "Hola Mundo"
print("startswith ", cadena.startswith("h"))
print("endswith ", cadena.endswith("a"))
print("split ", cadena.split(" "))
textoajuntar = ("1", "2", "3")
print("join ", "-".join(textoajuntar))

print("strip <", "  - - - -    ".strip() + ">")
print("replace ", cadena.replace("o", "ó"))
```

```
startswith  False
endswith  False
split  ['Hola', 'Mundo']
join  1-2-3
strip < - - - ->
replace  Hóla Mundó
```

Cadenas de caracteres




```
> s = "manzana"
> s.
```




```
Type "copyright", "credits" or "license()" for more information.
>>> a = "la manzana roja es la más sabrosa"
>>> a.count("la",0,25)
2
>>> x = "elefante"
>>> x.count("e",-1,-8)
0
>>> x = "elefante"
>>> x.count("e",-8,-1)
2
>>> x = "elefante"
>>> x.count("e",2,6)
1
>>> y = "¡Mmm! ¡Qué sabrosa está la manzana!"
>>> y.count("¡",0,100)
2
>>>
```

substring con dos caracteres literales

no cuenta de atrás hacia adelante

... pero de adelante hacia atrás con negativos, sí

no importa que exceda el índice final del largo de la string

Algunas operaciones con cadenas

```
>>> s1 = 'Hola'
>>> len(s1) # la función len() retorna la longitud de la cadena
4
>>> s2 = 'Quiero comer un pastel de chocolate'
>>> s2.split()
['Quiero', 'comer', 'un', 'pastel', 'de', 'chocolate']
```

```
>>> x = 'Hola'
>>> x.startswith('h')
False
>>> x.startswith('H')
True
>>> x.endswith('a')
True
>>> x.endswith('A')
False
>>> x.lower().startswith('h')
True
>>> x.lower().endswith('a')
True
```

```
>>> 'Hola' + ' a ' + 'todos'
'Hola a todos'
>>> 'xyz' * 5
'xyzxyzxyzxyzxyz'
>>> 5 * 'xyz'
'xyzxyzxyzxyzxyz'
>>> int('123')
123
```

Practica R01 – Métodos de las cadenas

POR ÚLTIMO, LA FORMA MÁS RECOMENDABLE Y PYTHÓNICA, PERO MÁS COMPLEJA, SERÍA USAR MATPLOTLIB MEDIANTE LA INTERFAZ ORIENTADA A OBJETOS. CUANDO SE PROGRAMA CON MATPLOTLIB, NO MIENTRAS SE TRABAJA INTERACTIVAMENTE, ESTA ES LA FORMA QUE
PERMITE TENER MÁS CONTROL SOBRE EL CÓDIGO. QUIZÁ VEAMOS ESTO EN EL FUTURO SI OS ANIMÁIS A ESCRIBIR SOBRE ELLO.

2. Muestra el número de veces que aparece cada una de las vocales.

Mercè Ribas Font

MERCE RIBAS FONT

MERCÈ RIBAS FONT

Mercè RIBAS FONT

Mercé RIBAS FONT

Métodos de las listas

Tabla de contenidos

append()

Añade un ítem al final de la lista:

Código Resultado

```
lista = [1,2,3,4,5]
lista.append(6)
lista
```

clear()

Vacía todos los ítems de una lista:

Código Resultado

```
lista.clear()
lista
```

https://docs.hektorprofe.net/python/metodos-de-las-colecciones/metodos-de-las-listas/

Practica R02 – Métodos de las listas

https://aprendeconalf.es/docencia/python/manual/diccionarios/