

Creado por:

Isabel Maniega

-1.1- Introducción a Python (Continuación)

-1.1.0- Diccionarios

```
In [1]: # clave-valor  
# "key": value  
# {"key": value}  
# {"key1": "value1", "key2": "value2",.....}
```

```
In [2]: diccionario = {"A": 10, "B": -2, "C": 35}  
diccionario
```

```
Out[2]: {'A': 10, 'B': -2, 'C': 35}
```

```
In [3]: diccionario["A"] # 10
```

```
Out[3]: 10
```

```
In [4]: diccionario["C"] # 35
```

```
Out[4]: 35
```

```
In [5]: diccionario["B"] # -2
```

```
Out[5]: -2
```

```
In [6]: len(diccionario)
```

```
Out[6]: 3
```

```
In [7]: diccionario = {"clave1": 1, "clave2": 2, "clave3": 3}  
diccionario
```

```
Out[7]: {'clave1': 1, 'clave2': 2, 'clave3': 3}
```

```
In [8]: # Mostrar valores de claves  
diccionario.keys()
```

```
Out[8]: dict_keys(['clave1', 'clave2', 'clave3'])
```

```
In [9]: type(diccionario.keys())
```

```
Out[9]: dict_keys
```

```
In [10]: # Mostrar valores de los valores  
diccionario.values()
```

```
Out[10]: dict_values([1, 2, 3])
```

```
In [11]: # Mostrar información del diccionario  
# Usada en Bucles  
diccionario.items()
```

```
Out[11]: dict_items([('clave1', 1), ('clave2', 2), ('clave3', 3)])
```

```
In [12]: # Modificación de un valor  
diccionario["clave1"] = 5  
diccionario
```

```
Out[12]: {'clave1': 5, 'clave2': 2, 'clave3': 3}
```

```
In [13]: # Eliminación de un campo del diccionario  
del diccionario["clave3"]  
diccionario
```

```
Out[13]: {'clave1': 5, 'clave2': 2}
```

```
In [14]: len(diccionario)
```

```
Out[14]: 2
```

Usos de tuplas/listas y diccionarios en Dataframe

```
In [15]: import pandas as pd
```

```
In [16]: # Usamos comillas dobles para los nombres de los estudiantes (E)  
E = ["Andres", "Marcos", "Eva", "María"]  
E
```

```
Out[16]: ['Andres', 'Marcos', 'Eva', 'María']
```

```
In [17]: # Notas de los exámenes (N), de 0 a 10, siendo 10 la nota más alta  
N = [9, 7, 8, 6]  
N
```

```
Out[17]: [9, 7, 8, 6]
```

```
In [18]: data = list(zip(E, N))  
data
```

```
Out[18]: [('Andres', 9), ('Marcos', 7), ('Eva', 8), ('María', 6)]
```

```
In [19]: df = pd.DataFrame(data, columns=["Estudiantes", "Notas"])  
df
```

Out[19]:

	Estudiantes	Notas
0	Andres	9
1	Marcos	7
2	Eva	8
3	María	6

```
In [20]: diccionario = {"Estudiantes": E, "Notas": N}
df2 = pd.DataFrame(diccionario)
df2
```

Out[20]:

	Estudiantes	Notas
0	Andres	9
1	Marcos	7
2	Eva	8
3	María	6

```
In [21]: curso = dict(Estudiantes=E, Notas=N)
curso
```

Out[21]: {'Estudiantes': ['Andres', 'Marcos', 'Eva', 'María'], 'Notas': [9, 7, 8, 6]}

-1.1.1- Strings

```
In [22]: s1 = "Hola, ¿Como estás?"
s1
```

Out[22]: 'Hola, ¿Como estás?'

```
In [23]: s1[-1], s1[17]
```

Out[23]: ('?', '?')

```
In [24]: len(s1)
```

Out[24]: 18

```
In [25]: s1[0], s1[1], s1[2], s1[3], s1[4], s1[5], s1[6], s1[7], s1[8], s1[9]
```

Out[25]: ('H', 'o', 'l', 'a', ',', ' ', '¿', 'C', 'o', 'm')

```
In [26]: s1[0]
```

Out[26]: 'H'

```
In [27]: # Ejemplo de lista para buscar palabras que empiezen por "J": ["Maria", '
# startswith = False, True (Booleano)
s1.startswith("J")
```

Out[27]: False

```
In [28]: s1.startswith("H")
```

Out[28]: True

```
In [29]: # Ejemplo de lista para buscar palabras que acaben por "J": ["Maria", "Ju  
# endswith = False, True (Booleano)  
s1.endswith("J")
```

Out[29]: False

```
In [30]: s1.endswith("?")
```

Out[30]: True

-1.1.2- Listas con falta de valores (missing values)

```
In [31]: L = [10, -20, None, 80, -5, None, 20]  
L
```

Out[31]: [10, -20, None, 80, -5, None, 20]

```
In [32]: L[2] = -1  
L
```

Out[32]: [10, -20, -1, 80, -5, None, 20]

```
In [33]: L[-2] = -1  
L
```

Out[33]: [10, -20, -1, 80, -5, -1, 20]

```
In [34]: # Bucles "for" muestre en lista  
for i in L:  
    print(i)
```

```
10  
-20  
-1  
80  
-5  
-1  
20
```

```
In [35]: L = [10, -20, None, 80, -5, None, 20]  
L
```

Out[35]: [10, -20, None, 80, -5, None, 20]

```
In [36]: # range: limita los valores a mostrar.  
# range decir empieza en posición 0 y acaba en posición 7  
for i in range(0, len(L)):  
    print(i)
```

0
1
2
3
4
5
6

```
In [37]: # condiciones: si esto es igual a x entonces...  
# if i == None:  
# sino haz esto otro = else:
```

```
In [38]: for i in range(0, len(L)):  
    if L[i] == None:  
        print(True)  
    else:  
        print(False)
```

False
False
True
False
False
True
False

```
In [39]: for i in range(0, len(L)):  
    if L[i] == None:  
        L[i] = -1  
L
```

Out[39]: [10, -20, -1, 80, -5, -1, 20]

Faltan valores en un Dataframe

```
In [40]: L = [10, -20, None, 80, -5, None, 20]  
L
```

Out[40]: [10, -20, None, 80, -5, None, 20]

```
In [41]: import pandas as pd  
df = pd.DataFrame(L, columns=["Temperatura"])  
df
```

Out[41]:

	Temperatura
0	10.0
1	-20.0
2	NaN
3	80.0
4	-5.0
5	NaN
6	20.0

```
In [42]: df.isnull()
```

```
Out[42]:
```

	Temperatura
0	False
1	False
2	True
3	False
4	False
5	True
6	False

```
In [43]: df.isnull().sum()
```

```
Out[43]: Temperatura      2  
dtype: int64
```

```
In [44]: df.describe()
```

```
Out[44]:
```

	Temperatura
count	5.000000
mean	17.000000
std	38.340579
min	-20.000000
25%	-5.000000
50%	10.000000
75%	20.000000
max	80.000000

```
In [45]: df
```

```
Out[45]:
```

	Temperatura
0	10.0
1	-20.0
2	NaN
3	80.0
4	-5.0
5	NaN
6	20.0

```
In [46]: df.Temperatura = df.Temperatura.fillna(df.Temperatura.mean())  
df
```

Out[46]: **Temperatura**

0	10.0
1	-20.0
2	17.0
3	80.0
4	-5.0
5	17.0
6	20.0

```
In [47]: df['Humedad'] = [10, 20, 30, 40, 50, 60, 70]
df
```

Out[47]: **Temperatura Humedad**

0	10.0	10
1	-20.0	20
2	17.0	30
3	80.0	40
4	-5.0	50
5	17.0	60
6	20.0	70

```
In [48]: df = df.drop(["Humedad"], axis=1)
df
```

Out[48]: **Temperatura**

0	10.0
1	-20.0
2	17.0
3	80.0
4	-5.0
5	17.0
6	20.0

Range de números

```
In [49]: # range(inicio, fin+1, salto)
# rango = range(1, 10) --> defecto el salto 1
rango = range(1, 10, 1)
rango
```

Out[49]: range(1, 10)

```
In [50]: # imprimimos los valores del rango
for numero in rango:
    print(numero)
```

1
2
3
4
5
6
7
8
9

```
In [51]: # les almacenamos e imprimos
listado_rango = []
for numero in rango:
    listado_rango.append(numero)
    # print("Añadiendo valores: ", listado_rango)
print("Listado final: ", listado_rango)
```

Listado final: [1, 2, 3, 4, 5, 6, 7, 8, 9]

np.arange: otra forma posible

```
In [52]: import numpy as np
```

```
In [53]: # range(inicio, fin+1, salto)
# rango = range(1, 10, 1)
rango_np = np.arange(1, 10, 1)
rango_np
```

Out[53]: array([1, 2, 3, 4, 5, 6, 7, 8, 9])

```
In [54]: # Convertir de array a lista
rango_lista = rango_np.tolist()
rango_lista
```

Out[54]: [1, 2, 3, 4, 5, 6, 7, 8, 9]

```
In [55]: # Aplicar saltos
# range(inicio, fin+1, salto)
num = np.arange(3, 37, 3).tolist()
num
```

Out[55]: [3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36]

```
In [56]: # Cuando es de 1 en 1. No es necesario añadirlo:
# rango = range(1, 10, 1)
rango_np = np.arange(1, 10)
rango_np
```

Out[56]: array([1, 2, 3, 4, 5, 6, 7, 8, 9])

```
In [57]: # si nosotros queremos que empiece 0, no es necesario indicarlo:
rango_np2 = np.arange(10)
rango_np2
```



```
Out[57]: array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

```
In [58]: # equivale a:  
# si nosotros queremos que empiece 0, no es necesario indicarlo:  
rango_np2 = np.arange(0, 10)  
rango_np2
```

```
Out[58]: array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

-1.2- Funciones, Clases y Objetos

Concepto de función general:

```
In [59]: # Funciones se definen:  
# def nombre de la funcion():
```

```
In [60]: # 1º definir función:  
def funcion_suma(x):  
    y = x + 20  
    return y
```

```
In [61]: funcion_suma(5)
```

```
Out[61]: 25
```

```
In [62]: funcion_suma(10)
```

```
Out[62]: 30
```

Función lambda

```
In [63]: # con funciones:  
def funcion_suma_1(x):  
    return x + 20  
funcion_suma_1(5)
```

```
Out[63]: 25
```

```
In [64]: funcion_suma_1(10)
```

```
Out[64]: 30
```

```
In [65]: # con lambda:  
(lambda x: x + 20)(5)
```

```
Out[65]: 25
```

```
In [66]: # con lambda:  
(lambda x: x + 20)(10)
```

```
Out[66]: 30
```

Break (continue, pass, NO EXPLICADOS de momento)

```
In [67]: L = [2, 5, 7, 9, 12, 25, -6]
L
```

```
Out[67]: [2, 5, 7, 9, 12, 25, -6]
```

```
In [68]: for numero in L:
          if numero == 9:
              print("\n")
              break # Rompa el bucles: pare el bucle for
          else:
              print(numero)
print("hemos llegado al valor 9, y salió del bucle FOR")
# 2, 5, 7 Les imprime
# 9 Salta (NO IMPRIME), \n permite dejar la linea en blanco
# 12, 25, -6 NO se IMPRIMEN
```

```
2
5
7
```

hemos llegado al valor 9, y salió del bucle FOR

Clases

```
In [69]: # definir una clase pone class + nombre de la clase:
class Python:
    def funcion_imprimir(y):
        print("estamos aquí en la función imprimir: ", y)

    def funcion_suma():
        x = 2
        z = 4
        y = x + z
        print("Estamos en la función suma y la suma de y es %s" %(y))
```

```
In [70]: Python.funcion_imprimir(8)
```

```
estamos aquí en la función imprimir: 8
```

```
In [71]: Python.funcion_suma()
```

```
Estamos en la función suma y la suma de y es 6
```

Programación Orientada a Objetos (POO)

```
In [72]: class Empleado:
          # funciones se les llama MÉTODOS.
          # variables se les llama ATRIBUTOS
          # Init se pone con (__) dos barras bajas:
          def __init__(self, Id, Name, Age, Role):
              self.Id = Id
              self.Name = Name
              self.Age = Age
              self.Role = Role
          # Instancio
```

```
# genero tantos clientes/empleados como quiera de esta forma  
# empleado1 es el objeto1  
empleado1 = Empleado(1, "Ana", 30, "Ingeniera")  
empleado2 = Empleado(2, "Pedro", 26, "Arquitecto")  
empleado3 = Empleado(3, "Maria", 54, "Abogado")
```

In [73]: empleado1.Age

Out[73]: 30

In [74]: empleado3.Role

Out[74]: 'Abogado'

Try - Except

In [75]: x = [1, 2, 3, 4]
x

Out[75]: [1, 2, 3, 4]

In [76]: **try:**
 print(w)
except:
 print("w no esta definida, no podemos imprimirla")

w no esta definida, no podemos imprimirla

In [77]: **try:**
 print(x)
except:
 print("w no esta definida, no podemos imprimirla")

[1, 2, 3, 4]

In [78]: **try:**
 print(w)
except Exception **as** e:
 print("### Error: %s" %str(e))

Error: name 'w' is not defined

Contenido creado por:

Isabel Maniega