**Iteración sobre Dictionaries** -{} curly braces -

x = {'Fede': 'fede@gmail.com', 'Bill' : 'billg@micro.com', 'Tom': None}

type(x)

🡪dict

Iterar sobre los nombres/keys:

for item in x.keys():

print(item)

Fede

Bill

Tom

Iterar sobre los valores(values) de los keys:

for item in x.values():

print(item)

fede@gmail.com

billg@micro.com

None

Iterar sobre los keys y values al mismo tiempo:

for key, value in x.items():

print(key, value)

Fede fede@gmail.com

Bill billg@micro.com

Tom None

Desempacar tuplas:

x = ('Fede', 'Bill' , 'Tom')

alumno1, alumno2, alumno3 = x

OJO! Nro de variales y elementos dentro de la tupla debe ser el mismo

alumno1

Out[20]: 'Fede'

alumno2

Out[21]: 'Bill'

alumno3

Out[22]: 'Tom'

**Strings formatting:**

Texto en Python utiliza Unicode (UTF) por defecto (incluye millones de símbolos).

Forma 1 “New Style” String Formatting (str.format) – Desde Python 3:

name = ‘Bob’

print('Hello, {}'.format(name))

'Hello, Bob'

-------- otro ejemplo ---------

sales\_record = {'price': 3.24,

'num\_items': 4,

'person': 'Chris'}

sales\_statement = '{} bought {} item(s) at a price of {} each for a total of {}'

print(sales\_statement.format(sales\_record['person'],

sales\_record['num\_items'],

sales\_record['price'],

sales\_record['num\_items']\*sales\_record['price']))

Para el string que contiene {} se utiliza el método format y se enumeran en orden las variables a imprimir. Estás variables se convierten a str automáticamente

Forma 2 String Interpolation / f-Strings – Desde Python 3.6

name = ‘Bob’

print(f'Hello, {name}!')

'Hello, Bob!'