In [2]:

```
l1 = 229
id (l1) #identificador
```

Out[2]:

```
1955711051312
```

In [3]:

```
l2 = 10
id (l2)
```

Out[3]:

```
1955710855760
```

In [4]:

```
l1 is l2 #No es lo mismo en valores
```

Out[4]:

```
False
```

In [6]:

```
a = 5

a is a
```

Out[6]:

```
True
```

In [7]:

```
id (a)
```

Out[7]:

```
1955710855600
```

In [18]:

```
#Cuando hablamos de valor puede coincidir el valor
#Cuando hablamos de variables no coinciden

g = 5 #De cero a 256 tienen el mismo identificador, a partir del 257 tienen distinto iden
tificador
h = 5

print (h == g)
print (h is g)
print (id(g))
print (id(g))
```

```
True
True
1955710855600
1955710855600
```

In [19]:

```
d = 875757

s = 875757

print (d == s)
```

```
print (d is s)
print(id (d))
print(id (s))
```

```
True
False
1955814372080
1955814372368
```

In [21]:

```
lista = [1, "dos", 3, "Pepito"]

print(lista[0])
print(lista[1])
print(lista[2])
print(lista[3])
```

```
1
dos
3
Pepito
```

In [23]:

```
for p in lista: #for + el nombre de una variable + in + colección (significado: por cada elemento en lista)
    print(p) #El primer valor de p es el primer elemento de la lista. Va a continuar con los valores hasta que se terminen los elementos
    print("----") #Pasa por todo el bucle hasta encontrar un nuevo elemento
```

```
1
----
dos
----
3
----
Pepito
----
```

In [25]:

```
for g in [2 , 4, 6, 8]:
    break #No muestra nada ya que ha roto el bucle directamente
```

In [26]:

```
for g in [2 , 4, 6, 8]:
    print(g)
    break
```

```
2
```

In [27]:

```
for g in [2 , 4, 6, 8]: #No muestra 6
    if g == 6:
        break
    print(g)
```

```
2
4
```

In [31]:

```
for g in [2 , 4, 6, 8]: #Muestra el 6 v1
    if g == 8:
        break
    print(g)
```

```
6
```

In [32]:

```
for g in [2 , 4, 6, 8]: #Muestra el 6 v2
    print(g)
    if g == 6:
        break
```

```
2
4
6
```

In [33]:

```
for g in [2 , 4, 6, 8]: #Muestra el 6 v3
    if g > 6:
        break
    print(g)
```

```
2
4
6
```

In [ ]:

```
for g in [2 , 4, 6, 8]: #Muestra el 6 solo porque el 2 al no ser igual, continua al sigu
iente, el 4 tampoco lo es, continua al siguiente, el 6 es igual por tanto imprime por tan
to lo rompe
    if g == 6:
        print(g)
        break
```

In [35]:

```
file1 = "Primera Fila", ["Maria", "Angeles", "Juan"]
file2 = "Segunda Fila", ["Marta", "Daniel", "Leo", "Miguel1", "Estela"]
file3 = "Tercera Fila", ["Kapil", "Roberto", "Alfonso", "Miguel2"]
fileR = "Remoto", ["Mar", "Alex", "Anais", "Antonio", "Ariadna", "Javi", "JaviOlcoz"]

print(file1[1][1]) #Para coger a Angeles
```

```
Angeles
```

In [39]:

```
altura1 = [1.78, 1.63, 1.75, 1.68]
altura2 = [2.00, 1.82]
altura3 = [1.65, 1.73, 1.75]
altura4 = [1.72, 1.71, 1.71, 1.62]

lista_alturas = [altura1, altura2, altura3, altura4]

print(lista_alturas[0][1])

x = [1.78, 1.63, 1.75, 1.68]
for x in lista_alturas:
    print(x[3])
```

```
1.63
1.68

---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-39-bcf63354e294> in <module>
     10 x = [1.78, 1.63, 1.75, 1.68]
     11 for x in lista_alturas:
---> 12     print(x[3])

IndexError: list index out of range
```

In [41]:

```
altura1 = [1.78, 1.63, 1.75, 1.68] #Queremos que se muestre el 1.68 y el 1.62
```

```
altura2 = [2.00, 1.82]
altura3 = [1.65, 1.73, 1.75]
altura4 = [1.72, 1.71, 1.71, 1.62]

lista_alturas = [altura1, altura2, altura3, altura4]

print(lista_alturas[0][1])
#x = [1.78, 1.63, 1.75, 1.68]
for x in lista_alturas:
    if len(x) > 3: #La lista ha de tener 4 elementos o más
       print(x[3])
```

```
1.63
1.68
1.62
```

In [43]:

```
print(range(6)) #For trabaja con valores, range trabaja con posiciones
type(range(6))
```

```
range(0, 6)
```

Out[43]:

```
range
```

In [44]:

```
print(list(range(6)))
```

```
[0, 1, 2, 3, 4, 5]
```

In [45]:

```
#Es lo mismo si:
r = list(range(6))
for i in r:
    print(r) #
```

```
[0, 1, 2, 3, 4, 5]
[0, 1, 2, 3, 4, 5]
[0, 1, 2, 3, 4, 5]
[0, 1, 2, 3, 4, 5]
[0, 1, 2, 3, 4, 5]
[0, 1, 2, 3, 4, 5]
```

In [42]:

```
for i in range(6): #Ejecuta 6 iteraciones como 6 recorridos
    print(i)
```

```
0
1
2
3
4
5
```

In [ ]:

```
lista = ["juan", "pepito", "silvia", "6", 7] #Para recorrer un rango en vez de una lista

for i in range(len(lista)):
    print(lista[i])
```

In [ ]:

```
lista = ["juan", "pepito", "silvia", "6", 7] #Para recorrer un rango en vez de una lista

tamano = len(lista)
for i in range(len(lista)):
```

```
    print(lista[i])
```

In [ ]:

```
#Con range creamos una colección que empieza en 0 hasta un número N. En ese caso, trabaja
mos con posiciones
```

In [47]:

```
#enumerate devuelve siempre dos valores, si pones dos variables te devuelve cada valor en
una variable. En cambio si pones solo una te devuelve 2 valores en una variable.
tupla= ("juan", "pepito", "silvia", "6", 7)
for posicion, valor in enumerate(tupla):
    print("position:", posicion)
    print("valor:", valor)
    print("-------")
```

```
position: 0
valor: juan
-------
position: 1
valor: pepito
-------
position: 2
valor: silvia
-------
position: 3
valor: 6
-------
position: 4
valor: 7
-------
```

In [50]:

```
#Para no mostrar silvia
tupla = ("juan", "pepito", "silvia", "6", 7)
for posicion, valor in enumerate(tupla):
    if valor != "silvia":
        print("position:", posicion)
        print("valor:", valor)
        print("-------")
```

```
position: 0
valor: juan
-------
position: 1
valor: pepito
-------
position: 3
valor: 6
-------
position: 4
valor: 7
-------
```

In [51]:

```
#Dos valores en una variable
tupla = ("juan", "pepito", "silvia", "6", 7)
for k in enumerate(tupla):
    print(k)
```

```
(0, 'juan')
(1, 'pepito')
(2, 'silvia')
(3, '6')
(4, 7)
```

In [57]:

```
#continue
```

```
tupla = ("juan", "pepito", "silvia", "6", 7)

for x in tupla:
    if x == "silvia":
        continue #cuando sea igual a silvia que salte a la siguiente iteración
    print(x)
```

```
juan
pepito
6
7
```

In [59]:

```
#for recorre una colección
#range es un numero, recorre el numero de iteraciones que va a tener nuestro for
for i in range(4): #bucle con 4 iteraciones
    s = input("Introduce una letra")
    print (s)
```

```
H
o
l
a
```

In [69]:

```
tupla1 = ("juan", "pepito", "silvia", "6", 7) #Que en la ultima vuelta no muestre silvia

for i in range (4): #lo que haya debajo se ejecuta 4 veces OJO el rango es 4 (4 repiticio
nes) pero la iteracion es 3 porque empieza a contar desde 0
    for x in tupla1: #Recorre cada valor de tupla y x tiene cada valor del elemento que
se está recorriendo
        if x == "silvia" and i == (3): #Cuando x sea silvia y cuando sea la ultima itera
cion
            continue
        print(x)
```

```
juan
pepito
silvia
6
7
juan
pepito
silvia
6
7
juan
pepito
silvia
6
7
juan
pepito
6
7
```

In [70]:

```
tupla = ("juan", "pepito", "silvia", "6", 7) #Que en la ultima vuelta no muestre silvia
for i in range (4): #lo que haya debajo se ejecuta 4 veces OJO el rango es 4 (4 repiticio
nes) pero la iteracion es 3 porque empieza a contar desde 0
    for x in tupla: #Recorre cada valor de tupla y x tiene cada valor del elemento que se
está recorriendo
        if x == "silvia" or i == 3: #Cuando x sea silvia y cuando sea la ultima iteracio
n
            continue
        print(x)
```

```
juan
pepito
```

```
6
7
juan
pepito
6
7
juan
pepito
6
7
```

In [71]:

```
lista = ["a", "b", "c", "d"]

for i in range(len(lista)):
    print(i) #Esta imprimiendo la longitud de la lista hasta ese range
```

```
0
1
2
3
```

In [72]:

```
lista = ["a", "b", "c", "d"] #Para acceder a cada elemento

for i in range(len(lista)):
    print(i, lista[i])
```

```
0 a
1 b
2 c
3 d
```

In [76]:

```
lista = ["a", "b", "c", "d"]
for i, x in enumerate(lista):
    print(i, x)
```

```
0 a
1 b
2 c
3 d
```

In [77]:

```
lista = ["a", "b", "c", "d"]
for i, x in enumerate(lista):
    print(i, lista[i])
```

```
0 a
1 b
2 c
3 d
```

In [80]:

```
acum = 0
for x in lista: #el valor del elemento
    print(acum, x)
    acum += 1 #Muestra la posición del elemento
```

```
0 a
1 b
2 c
3 d
```

In [81]:

```
acum = 0
```

```
for x in lista: #el valor del elemento
    acum += 1 #Si queremos que represente la posición esto no es válido, ya que las posic
iones empiezan en 0
    print(acum, x)
```

```
1 a
2 b
3 c
4 d
```

In [82]:

```
acum = -1
for x in lista: #el valor del elemento
    acum += 1 #Si lo ponemos en -1 si que valdría para mostrar la posición ya que empezam
os por uno menos
    print(acum, x)
```

```
0 a
1 b
2 c
3 d
```

In [1]:

```
print( (3**2)//2 )
```

```
4
```

In [ ]:

```
```