

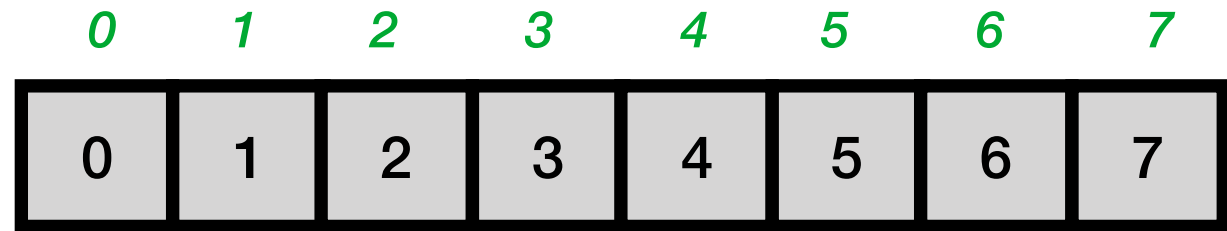
Sesión 3

Listas, tuplas y diccionarios

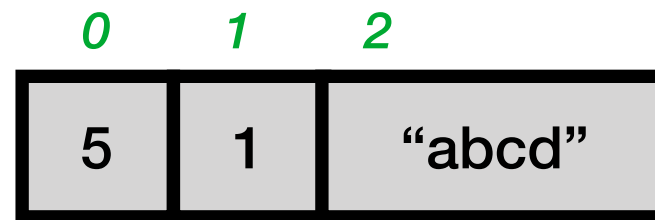
Rafael Cabañas de Paz
Curso: Introducción a Python
Almería, 9 Abril 2019



```
x = [0, 1, 2, 3, 4, 5, 6, 7]
```



```
y = [5, 1, "abcd"]
```



```
z = []
```

```
In [2]: len(x)
```

```
Out[2]: 8
```

```
In [3]: len(y)
```

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

```
In [4]: len(z)
```

```
Out[4]: 0
```

```
In [5]: x[2]
```

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

```
In [6]: y[2]
```

```
Out[6]: 'abcd'
```

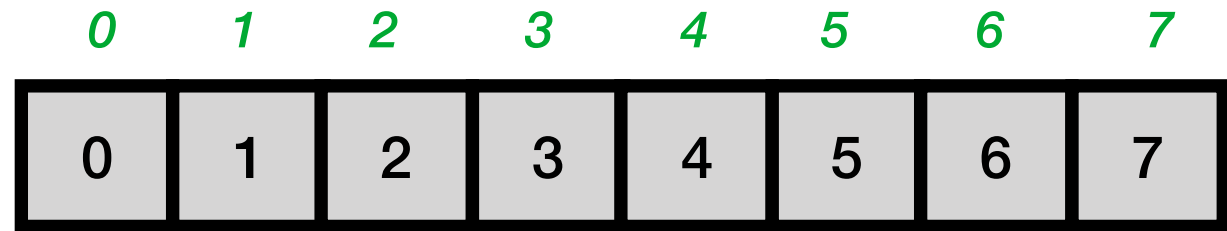
```
In [7]: x[-1]
```

```
Out[7]: 7
```

La función `len()` permite obtener la longitud

La **indexación** permite obtener el elemento en una determinada posición

```
x = [0,1,2,3,4,5,6,7]
```



- Hay 2 maneras de definir un bucle para definir una lista:

```
In [8]: for i in range(0, len(x)):  
        print(x[i])
```

0
1
2
3
4
5
6
7

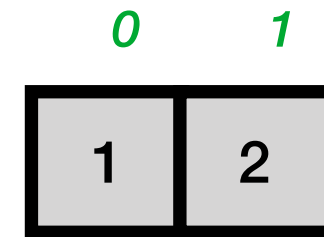
Recorriendo los índices

```
In [9]: for elem in x:  
        print(elem)
```

0
1
2
3
4
5
6
7

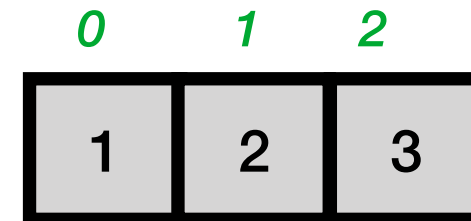
Recorriendo los elementos

```
x = [1, 2]
```

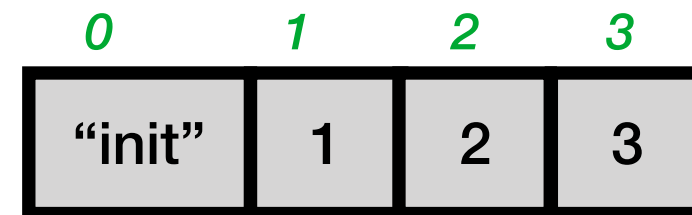


```
x.append(3)
```

elemento

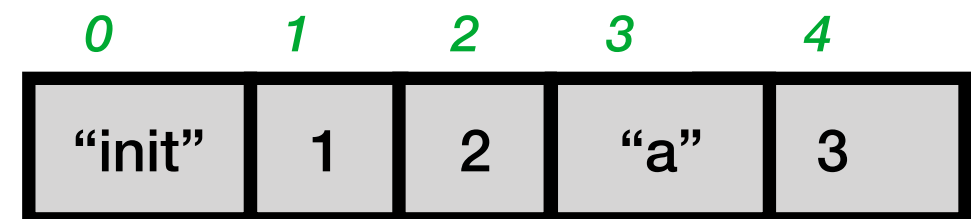


```
x.insert(0, "init")
```



```
x.insert(-1, "a")
```

posición elemento

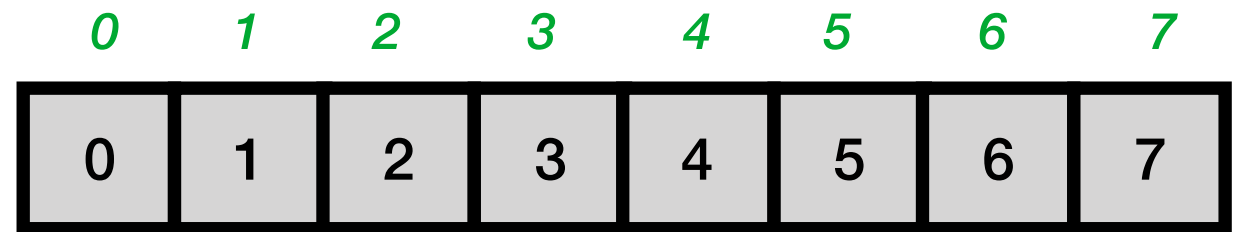


- **Ejercicio:** ¿Qué lista se imprime por pantalla?

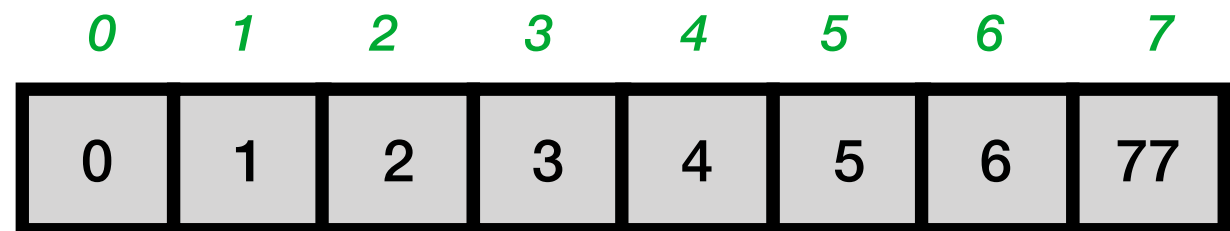
```
x = []  
x.append(9)  
x.insert(0,0)  
x.insert(0,2)  
x.insert(-1,1)  
  
print(x)
```

- Se pueden sobrescribir y borrar elementos

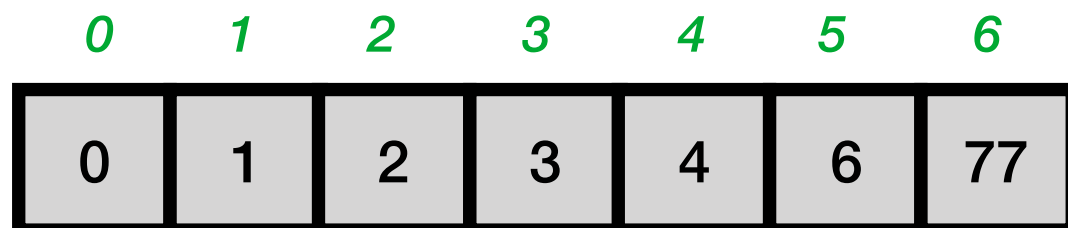
`x = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]`



`x[7] = 77`



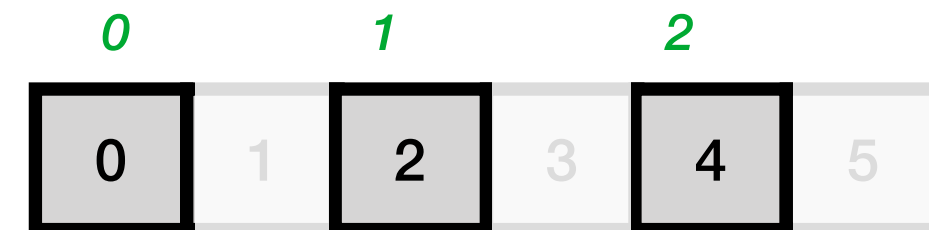
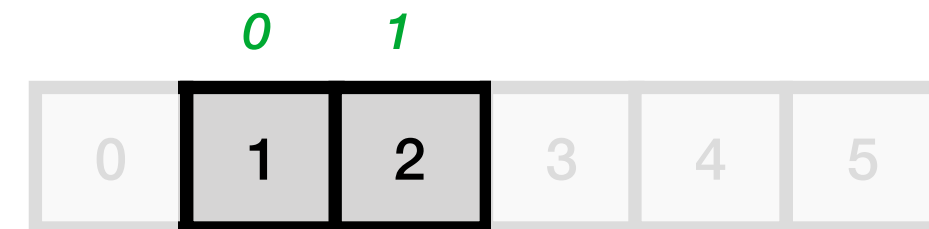
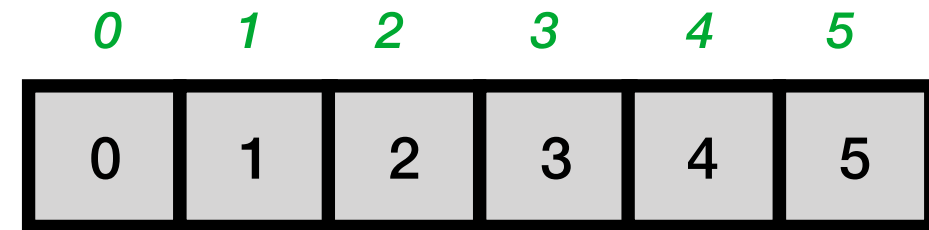
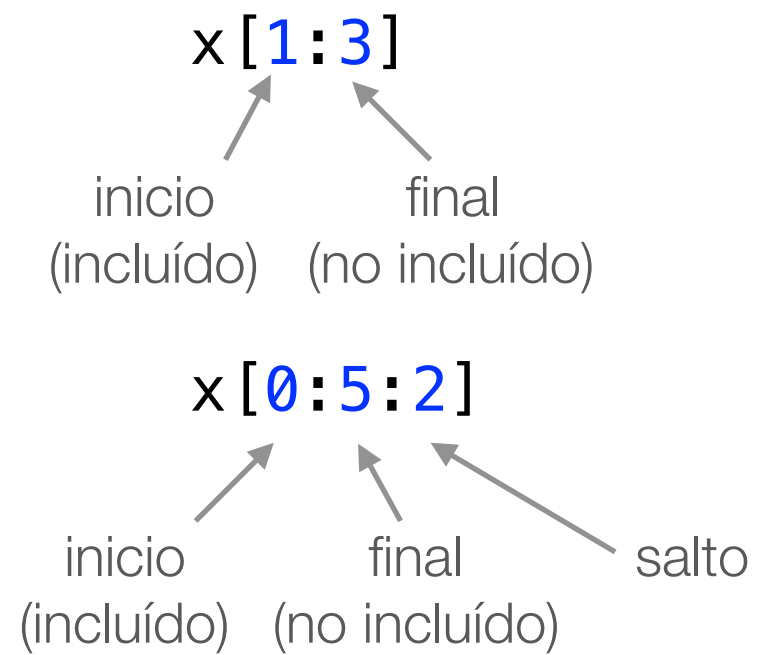
`del x[5]`



- Atención: el borrado de un elemento puede afectar a la posición que ocupan otros elementos de la lista

Slicing

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



- Los operadores `*` y `+` no son operadores aritméticos con listas

```
In [11]: [0,1] + [1,2]
```

Concatenación

```
Out[11]: [0, 1, 1, 2]
```

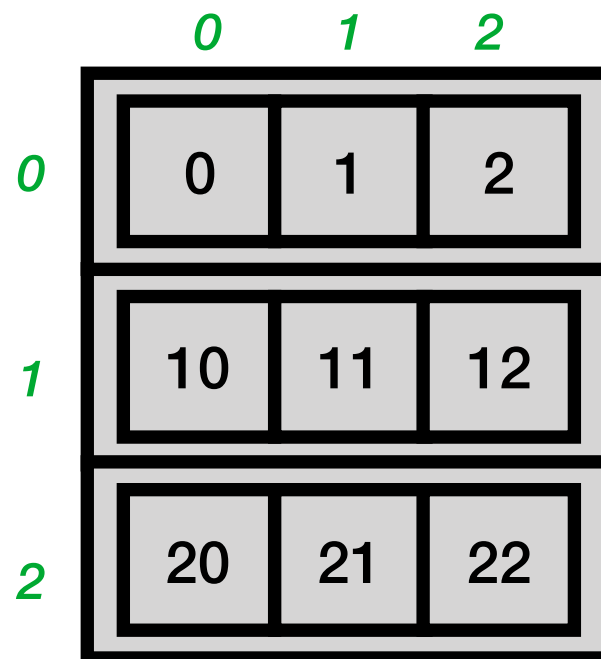
```
In [12]: [0,1,2] * 5
```

Repetición

```
Out[12]: [0, 1, 2, 0, 1, 2, 0, 1, 2]
```


- Una lista puede estar compuesta a por cualquier tipo de elemento:
incluso otras listas

```
x = [[0,1,2],[10,12,13],[20,21,22]]
```



```
x[1][0] == 10
```

```
len(x) == 3
```

```
x2 = []  
for elem in x:  
    x2.append(2*elem)
```



```
x2 = [2*elem for elem in x]
```

↑ ↑ ↑
expresión variable lista de
de salida entrada

```
x2 = []  
for elem in x:  
    if elem>3:  
        x2.append(2*elem)
```



```
x2 = [2*elem for elem in x if elem>3]
```

↑
predicado de
filtrado
(opcional)

- Una tupla es un objeto multidimensional **inmutable**

```
t = (1, 2)
```

- Admite la mayoría de operaciones que se pueden realizar sobre una lista

```
len(t)  
t[0:2]
```

- No se puede modificar

```
t[0] = 4
```

Error !!!

- Se pueden transformar las listas en tuplas y vice-versa

```
x = list(t)  
t = tuple(x)
```

- Cada elemento en un diccionario está compuesto por una clave (única) y un valor

```
temp_dict = {"ene": 8, "feb": 7, "mar": 10,  
             "abr": 16, "may": 18, "jun": 20,  
             "jul": 25, "ago": 27, "sept": 20,  
             "oct": 17, "nov": 14  
            }
```

- Indexación

```
temp_dict["feb"]
```

- Modificación

```
temp_dict.update({"dic": 11})
```

```
dict_vacio = {}
```