



## Funciones aplicables a listas

Al generar un objeto llamado `lista_de_numeros` que se compone de los números del 1 al 10, Python le concederá una serie de acciones dado que infiere que su mejor representación es una lista. Se puede ver cuáles son todas las posibles acciones, utilizando el atributo `__dir__()`.

```
lista_de_numeros = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
print(lista_de_numeros.__dir__())
```

```
['__repr__', '__hash__', '__getattribute__', '__lt__', '__le__', '__eq__', '__ne__',  
 '__gt__', '__ge__', '__iter__', '__init__', '__len__', '__getitem__', '__setitem__',  
 '__delitem__', '__add__', '__mul__', '__rmul__', '__contains__', '__iadd__', '__imul__',  
 '__new__', '__reversed__', '__sizeof__', 'clear', 'copy', 'append', 'insert', 'extend',  
 'pop', 'remove', 'index', 'count', 'reverse', 'sort', '__doc__', '__str__',  
 '__setattr__', '__delattr__', '__reduce_ex__', '__reduce__', '__subclasshook__',  
 '__init_subclass__', '__format__', '__dir__', '__class__']
```

Aquellos elementos que están envueltos por `__` se conocen como *magic built-in* o *dunders*, y buscan generar flexibilizaciones en el comportamiento de las clases. Serán retomados cuando hablemos de clases. Todas las que no son *dunders* son las acciones disponibles a acceder.