Métodos principales de diccionarios

```
dict1.clear       dict1.get         dict1.pop         dict1.update
dict1.copy        dict1.items       dict1.popitem     dict1.values
dict1.fromkeys    dict1.keys        dict1.setdefault
```

Métodos de eliminación: clear

```
>>> dict1 = dict(one=1, two=2, three=3)
>>> dict1.clear()
>>> dict1
{}
```

Métodos de agregado y creación: copy, dict.fromkeys, update, setdefault

```
>>> dict1 = dict(one=1, two=2, three=3)
>>> dict2 = dict1.copy()

>>> dict.fromkeys(["one","two","three"])
{'one': None, 'two': None, 'three': None}
>>> dict.fromkeys(["one","two","three"],100)
{'one': 100, 'two': 100, 'three': 100}

>>> dict1 = dict(one=1, two=2, three=3)
>>> dict2 = {'four':4,'five':5}
>>> dict1.update(dict2)
>>> dict1
{'one': 1, 'two': 2, 'three': 3, 'four': 4, 'five': 5}

>>> dict1 = dict(one=1, two=2, three=3)
>>> dict1.setdefault("four",4)
4
>>> dict1
{'one': 1, 'two': 2, 'three': 3, 'four': 4}
>>> dict1.setdefault("one",-1)
1
>>> dict1
{'one': 1, 'two': 2, 'three': 3, 'four': 4}
```

Métodos de retorno: get, pop, popitem, items, keys, values

```
>>> dict1 = dict(one=1, two=2, three=3)
>>> dict1.get("one")
1
>>> dict1.get("four")
>>> dict1.get("four","no existe")
'no existe'

>>> dict1.pop("one")
1
>>> dict1
{'two': 2, 'three': 3}
>>> dict1.pop("four")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 'four'
>>> dict1.pop("four","no existe")
'no existe'

>>> dict1 = dict(one=1, two=2, three=3)
>>> dict1.popitem()
('one', 1)
>>> dict1
{'two': 2, 'three': 3}

>>> dict1 = dict(one=1, two=2, three=3)
>>> dict1.items()
dict_items([('one', 1), ('two', 2), ('three', 3)])

>>> dict1.keys()
dict_keys(['one', 'two', 'three'])
```

El tipo de datos dictviews

Los tres últimos métodos devuelven un objeto de tipo `dictviews`.

Esto devuelve una vista dinámica del ciccionario, por ejemplo:

```
>>> dict1 = dict(one=1, two=2, three=3)
>>> i = dict1.items()
>>> i
dict_items([('one', 1), ('two', 2), ('three', 3)])
>>> dict1["four"]=4
>>> i
dict_items([('one', 1), ('two', 2), ('three', 3), ('four', 4)])
```

Es este tipo de datos podemos usar las siguientes funciones:

- `len()` : Devuelve número de elementos de la vista.
- `iter()` : Nos devuelve un iterador de las claves, valores o ambas.
- `x in dictview` : Devuelve True si x está en las claves o valores.

Recorrido de diccionarios

Podemos recorrer las claves:

```
>>> for clave in dict1.keys():
...     print(clave)
one
two
three
```

Podemos recorrer los valores:

```
>>> for valor in dict1.values():
...    print(valor)
1
2
3
```

O podemos recorrer ambos:

```
>>> for clave,valor in dict1.items():
...   print(clave,"->",valor)
one -> 1
two -> 2
three -> 3
```