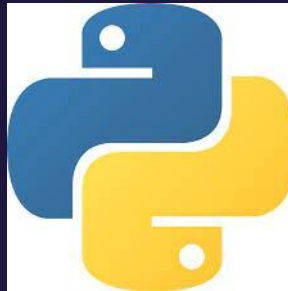


LENGUAJES y HERRAMIENTA PARA CIENCIAS DE DATOS I

Diccionarios I



¿Qué es un diccionario?

- Contenedor pares clave:valor
- Operaciones principales
 - ◆ Almacenar un valor asociado a un clave
 - ◆ Recuperar un valor a partir de una clave
- Características
 - ◆ Tipo mutable
 - ◆ Tipo ordenado



Crear un diccionario

- Secuencia de pares clave:valor entre {}
 - ◆ {clave1:valor1, clave2:valor2,}

```
mluque@hydrogen: ~  
File Edit View Search Terminal Help  
mluque@hydrogen:~$ python3  
Python 3.6.9 (default, Jan 26 2021, 15:33:00)  
[GCC 8.4.0] on linux  
Type "help", "copyright", "credits" or "license" for more information.  
>>> d = {'a':1, 'b':"hola", 'c': 3.12, 5:'a'}  
>>> d  
{'a': 1, 'b': 'hola', 'c': 3.12, 5: 'a'}  
>>> 
```

Crear un diccionario

- Usando el constructor de la clase (dict)

- ◆ Pares clave:valor entre {}

```
>>> d = dict({'a':1, 'b':2, 'c':3})  
>>> d  
{'a': 1, 'b': 2, 'c': 3}
```

- ◆ Argumentos con nombre

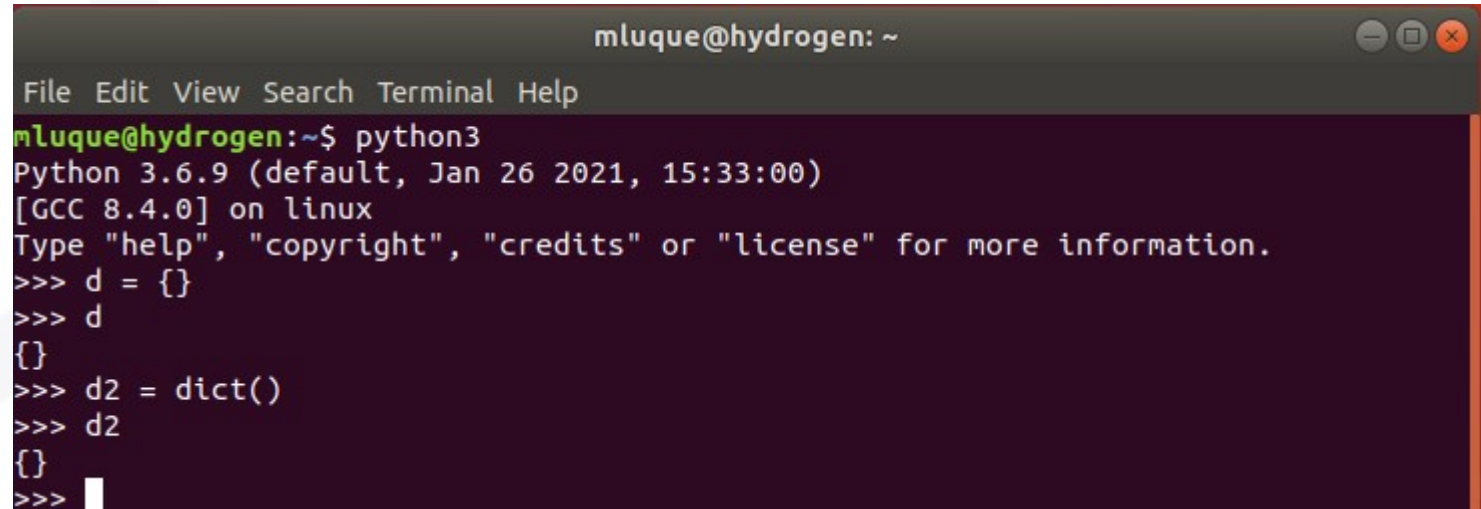
```
>>> d2 = dict(a=1, b=2, c=3)  
>>> d2  
{'a': 1, 'b': 2, 'c': 3}
```

- ◆ Pasando un iterable

```
>>> d3 = dict([('a',1), ('b',2), ('c',3)])  
>>> d3  
{'a': 1, 'b': 2, 'c': 3}
```

Crear un diccionario

- Casos especiales
 - ◆ Diccionario vacío
 - {}
 - dict()



```
mluque@hydrogen: ~  
File Edit View Search Terminal Help  
mluque@hydrogen:~$ python3  
Python 3.6.9 (default, Jan 26 2021, 15:33:00)  
[GCC 8.4.0] on linux  
Type "help", "copyright", "credits" or "license" for more information.  
>>> d = {}  
>>> d  
{}  
>>> d2 = dict()  
>>> d2  
{}  
>>>
```

Acceso a los elementos

- Indexación por clave

- ◆ `d[clave]`

```
>>> d = {'a':1, 'b':2, 'c':3, 'd':4}
>>> d
{'a': 1, 'b': 2, 'c': 3, 'd': 4}
>>> d['b']
2
>>> d['e']
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 'e'
```

- Método `get`

- ◆ `get(clave, [valor_por_defecto])`

```
>>> d.get('c')
3
>>> d.get('e')
>>> d.get('e', 'Incorrecta')
'Incorrecta'
```

Recorrer un diccionario

- Utilizar un bucle for
 - ◆ Recorrer los pares → `dict.items()`
 - ◆ Recorrer solo claves → `dict.keys()`
 - ◆ Recorrer solo valores → `dict.values()`

```
>>> d = {'a':1, 'b':2, 'c':3, 'd':4}
>>> for i in d.items():
...     print(i)
...
('a', 1)
('b', 2)
('c', 3)
('d', 4)
```

```
>>> for i in d.keys():
...     print(i)
...
a
b
c
d
```

```
>>> for i in d.values():
...     print(i)
...
1
2
3
4
>>>
```

Añadir/Modificar elementos

- `d[clave] = valor`
 - ◆ Si clave existe modifica su valor
 - ◆ Si clave no existe añade un nuevo par clave:valor

```
>>> d = {'a':1, 'b':2, 'c':3, 'd':4}
>>> d['e']=5
>>> d
{'a': 1, 'b': 2, 'c': 3, 'd': 4, 'e': 5}
>>> d['b']=7
>>> d
{'a': 1, 'b': 7, 'c': 3, 'd': 4, 'e': 5}
>>> 
```


Añadir elementos

- `setdefault(clave, [valor])`
 - ◆ Añade el par clave:valor o clave:None si valor no se especifica

```
>>> d = {'a':1, 'b':2, 'c':3, 'd':4}
>>> d.setdefault('e',5)
5
>>> d
{'a': 1, 'b': 2, 'c': 3, 'd': 4, 'e': 5}
>>> d.setdefault('f')
>>> d
{'a': 1, 'b': 2, 'c': 3, 'd': 4, 'e': 5, 'f': None}
>>>
```

Eliminar elementos

- `pop(clave, [valor_por_defecto])`
- `popitem()`
- `del`
- `clear()`

```
>>> d.popitem()
('d', 4)
>>> d
{'b': 2, 'c': 3}
>>> del d['b']
>>> d
{'c': 3}
>>> d.clear()
>>> d
{}
```

```
>>> d = {'a':1, 'b':2, 'c':3, 'd':4}
>>> d.pop('a')
1
>>> d
{'b': 2, 'c': 3, 'd': 4}
>>> d.pop('e', 'No existe')
'No existe'
>>> d.pop('e')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 'e'
```

