



Diccionarios

Python Básico | ⌚ 16 minutos

Aprenda como usar y modificar diccionarios.

Accesar datos

Como vimos en Tipos de Datos, podemos acceder a los elementos de un diccionario usando paréntesis cuadrados con su key adentro.

Este método va a retornar un error si el key no existe.

Para evitar esto, podemos usar el método `get`, que retornará `None` si el key no existe.

Veamos una comparación:

Acceso directo con índices

```
1 course_information = {  
2     'title': 'Introduction to DBs',  
3     'description': 'Here we review the basics of SQL Databases',  
4     'length_in_minutes': 600,  
5 }  
6  
7 print(course_information['description'])
```

Copiar



Here we review the basics of SQL Databases

Acceso usando `get`



None

Iteración

Iterando el items

```
1 europe_capitals_by_country = {  
2     'spain' : 'madrid',  
3     'france' : 'paris',  
4     'germany' : 'berlin',  
5     'norway' : 'oslo',  
6 }  
7  
8 for country, capital in europe_capitals_by_country.items():  
9     print(f'{country} : {capital}')
```

Copiar



```
france : paris  
spain : madrid  
normway : oslo  
germany : berlin
```

También podemos acceder solo a los `keys` o a los `values` de un diccionario usando el método llamado igual a estos.

Los diccionarios no tienen orden!

Iterando el keys

```
1 europe_capitals_by_country = {  
2     'spain' : 'madrid',  
3     'france' : 'paris',  
4     'germany' : 'berlin',  
5     'norway' : 'oslo',  
6 }  
7  
8 for country in europe_capitals_by_country.keys():  
9     print(country)
```

Copiar



spain
france
germany
norway

Iterando el values

```
1 europe_capitals_by_country = {  
2     'spain' : 'madrid',  
3     'france' : 'paris',  
4     'germany' : 'berlin',  
5     'norway' : 'oslo',  
6 }  
7  
8 for capital in europe_capitals_by_country.values():  
9     print(capital)
```

Copiar



madrid
paris
berlin
oslo

Agregar datos

Debido a que los datos de los diccionarios están atados a keys, debemos acceder a la key que queremos llenar (aunque no exista) y darle un valor:

```
1 user_data = {  
2     'full_name': 'John Snow',  
3     'email': 'j.snow@gmail.com',  
4 }  
5  
6 user_data['password'] = 'WinterIsComing2023'  
7 print(user_data)
```

Copiar



{'full_name': 'John Snow', 'email': 'j.snow@gmail.com', 'password':
'WinterIsComing2023'}

Eliminar datos

Al igual que con las listas, podemos usar el método `pop` para eliminar el elemento de un key específico:

Este método también retorna el elemento eliminado.

```
1 student_information = {
2     'first_name': 'Harry',
3     'last_name': 'Potter',
4     'age': 17,
5 }
6
7 deleted_item = student_information.pop('last_name')
8 print(student_information)
9 print(f'Deleted item: {deleted_item}')
```

Copiar



{'first_name': 'Harry', 'age': 17}

Deleted item: Potter