# Diccionarios en Python

Los diccionarios en Python son una estructura de datos que almacena pares de clave-valor, donde cada clave es única y se utiliza para acceder a su valor correspondiente. A continuación, se presentan las propiedades y métodos más importantes de los diccionarios en Python, junto con ejemplos de código.

## Propiedades de los Diccionarios

### 1. Acceso a elementos

Los elementos de un diccionario se acceden mediante sus claves, utilizando corchetes [].

persona = {"nombre": "Juan", "edad": 30, "ciudad": "Madrid"}  
print(persona["nombre"]) # Salida: "Juan"

### 2. Mutabilidad

Los diccionarios son mutables, lo que significa que sus elementos (valores) pueden ser modificados después de su creación.

persona = {"nombre": "Juan", "edad": 30, "ciudad": "Madrid"}  
persona["edad"] = 31 # Modifica el valor de la clave "edad"  
print(persona) # Salida: {"nombre": "Juan", "edad": 31, "ciudad": "Madrid"}

### 3. Longitud

La longitud de un diccionario (número de pares clave-valor) se puede obtener utilizando la función len().

persona = {"nombre": "Juan", "edad": 30, "ciudad": "Madrid"}  
print(len(persona)) # Salida: 3

## Métodos de los Diccionarios

### 1. get()

Devuelve el valor de una clave específica. Si la clave no existe, se puede proporcionar un valor predeterminado opcional.

persona = {"nombre": "Juan", "edad": 30, "ciudad": "Madrid"}  
print(persona.get("edad")) # Salida: 30  
print(persona.get("profesion", "Desconocida")) # Salida: "Desconocida"

### 2. keys()

Devuelve un objeto de vista que contiene todas las claves del diccionario.

persona = {"nombre": "Juan", "edad": 30, "ciudad": "Madrid"}  
claves = persona.keys()  
print(claves) # Salida: dict\_keys(["nombre", "edad", "ciudad"])

### 3. values()

Devuelve un objeto de vista que contiene todos los valores del diccionario.

persona = {"nombre": "Juan", "edad": 30, "ciudad": "Madrid"}  
valores = persona.values()  
print(valores) # Salida: dict\_values(["Juan", 30, "Madrid"])

### 4. items()

Devuelve un objeto de vista que contiene tuplas de pares clave-valor del diccionario.

persona = {"nombre": "Juan", "edad": 30, "ciudad": "Madrid"}  
items = persona.items()  
print(items) # Salida: dict\_items([("nombre", "Juan"), ("edad", 30), ("ciudad", "Madrid")])

### 5. update()

Actualiza el diccionario con los pares clave-valor de otro diccionario o un objeto iterable de pares clave-valor.

persona = {"nombre": "Juan", "edad": 30}  
datos\_adicionales = {"ciudad": "Madrid", "profesion": "Ingeniero"}  
persona.update(datos\_adicionales)  
print(persona) # Salida: {"nombre": "Juan", "edad": 30, "ciudad": "Madrid", "profesion": "Ingeniero"}

Los diccionarios son una estructura de datos muy poderosa y flexible en Python, ya que permiten almacenar y acceder a datos de una manera eficiente y organizada. Son ampliamente utilizados en diversos contextos, como el procesamiento de datos, la creación de bases de datos simples y la representación de objetos complejos.