

✓ Clase 08 - Diccionarios en Python

✓ 1. Diccionarios: Clave-Valor, Acceso, Inserción y Eliminación

Definición

Un diccionario es una colección de elementos que, a diferencia de las listas o tuplas, no se almacenan en un orden secuencial, sino que cada elemento tiene una **clave** única y un **valor** asociado. Piensa en un diccionario real: buscas una palabra (la clave) para encontrar su definición (el valor). En Python, los diccionarios nos permiten almacenar información de forma lógica y acceder a ella de manera muy eficiente.

Propiedades y Características

- **Mutable:** Podemos modificar, añadir o eliminar elementos después de su creación.
- **Dinámicos:** Pueden crecer o decrecer en tamaño según lo necesitemos.
- **Claves únicas e inmutables:** No puede haber dos claves iguales en un mismo diccionario. Además, las claves deben ser de un tipo de dato inmutable (como strings, números o tuplas).
- **Ordenados (a partir de Python 3.7):** Desde esta versión, los diccionarios recuerdan el orden en que se insertaron los elementos. En versiones anteriores, eran desordenados.

✓ Sintaxis

Se crean utilizando llaves `{}` y separando cada par clave-valor con dos puntos `:`.

```
1 # Creación de un diccionario
2 mi_diccionario = {
3     "clave1": "valor1",
4     "clave2": "valor2",
5     "clave3": "valor3"
6 }
```

```
1 mi_diccionario
```

```
{'clave1': 'valor1', 'clave2': 'valor2', 'clave3': 'valor3'}
```

```
1 # Creación de un diccionario vacío
2 diccionario_vacio = {}
3 diccionario_vacio
```

```
{}
```

```
1 otro_vacio = dict()
2 otro_vacio
```

```
{}
```

```
1 conversion = dict(['a1', 13, 'b4', 25, 'c8', 36])
2 conversion
```

```
-----
TypeError                                Traceback (most recent call last)
/tmp/ipython-input-391222313.py in <cell line: 0>()
----> 1 conversion = dict(['a1', 13, 'b4', 25, 'c8', 36])
      2 conversion
```

```
TypeError: cannot convert dictionary update sequence element #1 to a sequence
```

Pasos siguientes: [Explicar error](#)

```
1 diccionario_1= {'0123457L':['Pedro', 'Perez', 'Dirección']}
```

✓ Ejemplos Resueltos

```
1 # 1. Creación y acceso a un diccionario
2 estudiante = {
3     "nombre": "Ana",
4     "edad": 22,
5     "curso": "Inteligencia Artificial",
6     "es_becado": True
7 }
8
9 print(f"Diccionario original: {estudiante}")
```

Diccionario original: {'nombre': 'Ana', 'edad': 22, 'curso': 'Inteligencia Artificial', 'es_becado': True}

```
1 # Accedemos al valor asociado a la clave 'nombre'
2 edad_estudiante = estudiante["edad"]
3 print(f"El nombre del estudiante es: {edad_estudiante}")
```

El nombre del estudiante es: 22

```
1 # 2. Inserción y Modificación de elementos
2
3 # Añadimos un nuevo par clave-valor para la universidad
4 estudiante["universidad"] = "Universidad Politécnica"
5 print(f"Diccionario con nueva clave: {estudiante}")
```

Diccionario con nueva clave: {'nombre': 'Ana', 'edad': 22, 'curso': 'Inteligencia Artificial', 'es_becado': True, 'universidad': 'Universidad Politécnica'}

```
1 # Modificamos un valor existente. Ana ha cumplido años.
2 estudiante["edad"] = 23
3 print(f"Diccionario con edad actualizada: {estudiante}")
```

Diccionario con edad actualizada: {'nombre': 'Ana', 'edad': 23, 'curso': 'Inteligencia Artificial', 'es_becado': True, 'universidad': 'Universidad Politécnica'}

1 Empieza a programar o a [crear código](#) con IA.

```
1 # 3. Eliminación de elementos
2
3 # Usamos la palabra clave 'del' para eliminar el par con la clave 'es_becado'
4 del estudiante["es_becado"]
5 print(f"Diccionario tras eliminar una clave: {estudiante}")
```

Diccionario tras eliminar una clave: {'nombre': 'Ana', 'edad': 23, 'curso': 'Inteligencia Artificial', 'universidad': 'Universidad Politécnica'}

✓ Ejercicios Propuestos

1. **Crear un diccionario** llamado `mi_coche` que almacene la siguiente información: `marca` (string), `modelo` (string), `año` (integer) y `color` (string).
2. **Acceder y mostrar** por pantalla el modelo del coche.
3. **Añadir una nueva clave** llamada `kilometraje` con un valor numérico.
4. **Actualizar el valor** de la clave `color` a un nuevo color que te guste.
5. **Eliminar la clave** `año` del diccionario.

✓ 2. Métodos de Diccionarios

Los diccionarios vienen con métodos integrados que nos facilitan mucho el trabajo. Veamos los más importantes.

✓ `.keys()`

Devuelve un objeto `dict_keys` que contiene todas las claves del diccionario. Es muy útil para iterar sobre ellas.

✓ Sintaxis

```
diccionario.keys()
```

```
1 configuracion = {
2     "idioma": "español",
3     "tema": "oscuro",
4     "notificaciones": True
5 }
6
7 claves = configuracion.keys()
8 print(f"Claves del diccionario: {claves}")
9
10
```

```
Claves del diccionario: dict_keys(['idioma', 'tema', 'notificaciones'])
```

```
1 # Podemos convertirlo a lista si necesitamos
2 lista_claves = list(claves)
3 print(f"Claves como lista: {lista_claves}")
```

```
Claves como lista: ['idioma', 'tema', 'notificaciones']
```

Ejercicio Propuesto

- Dado un diccionario `producto = {"nombre": "Laptop", "precio": 1200, "stock": 45}`, obtén y muestra por pantalla todas sus claves.

✓ `.values()`

Devuelve un objeto `dict_values` con todos los valores del diccionario.

✓ Sintaxis

```
diccionario.values()
```

```
1 configuracion = {
2     "idioma": "español",
3     "tema": "oscuro",
4     "notificaciones": True
5 }
6
7 valores = configuracion.values()
8 print(f"Valores del diccionario: {valores}")
9
10
```

```
Valores del diccionario: dict_values(['español', 'oscuro', True])
```

```
1 # También podemos convertirlo a lista
2 lista_valores = list(valores)
3 print(f"Valores como lista: {lista_valores}")
```

```
Valores como lista: ['español', 'oscuro', True]
```

Ejercicio Propuesto

- Para el diccionario `producto` del ejercicio anterior, obtén y muestra todos sus valores.

✓ `.items()`

Devuelve un objeto `dict_items` que contiene tuplas, donde cada tupla es un par `(clave, valor)`.

✓ Sintaxis

```
diccionario.items()
```

```

1 configuracion = {
2     "idioma": "español",
3     "tema": "oscuro",
4     "notificaciones": True
5 }
6
7 items = configuracion.items()
8 print(f"Items del diccionario: {items}")
9
10

```

```
Items del diccionario: dict_items([('idioma', 'español'), ('tema', 'oscuro'), ('notificaciones', True)])
```

```

1 # Es especialmente útil para recorrer el diccionario en un bucle
2 print("\nRecorriendo el diccionario:")
3 for clave, valor in configuracion.items():
4     print(f"- {clave}: {valor}")

```

```

Recorriendo el diccionario:
- idioma: español
- tema: oscuro
- notificaciones: True

```

Ejercicio Propuesto

- Para el diccionario `producto`, obtén sus items y luego usa un bucle `for` para imprimir cada clave y su valor en una línea separada, por ejemplo: `nombre: Laptop`.

✓ `.get(clave, valor_por_defecto)`

Permite acceder a una clave de forma segura. Si la clave no existe, no lanza un error, sino que devuelve `None` o el valor por defecto que especifiquemos.

✓ Sintaxis

```

diccionario.get('clave_existente')
diccionario.get('clave_inexistente', 'valor si no se encuentra')

```

```

1 estudiante = {
2     "nombre": "Carlos",
3     "edad": 25
4 }
5
6 # Acceso a una clave que sí existe
7 nombre = estudiante.get("nombre")
8 print(f"Nombre obtenido con get: {nombre}")

```

```
Nombre obtenido con get: Carlos
```

```

1 # Intento de acceso a una clave que no existe (sin valor por defecto)
2 apellido = estudiante.get("apellido")
3 print(f"Apellido obtenido con get: {apellido}") # Devuelve None

```

```
Apellido obtenido con get: None
```

```

1 # Intento de acceso con un valor por defecto
2 curso = estudiante.get("curso", "No inscrito")
3 print(f"Curso obtenido con get: {curso}")

```

```
Curso obtenido con get: No inscrito
```

Ejercicio Propuesto

- Usando el diccionario `mi_coche` de antes, intenta obtener el valor de la clave `cv` (caballos de vapor). Como no existirá, proporciona un valor por defecto de `'Dato no disponible'` y muéstralo por pantalla.

✓ `.update(otro_diccionario)`

Fusiona un diccionario con otro. Si hay claves coincidentes, los valores del diccionario que se pasa como argumento sobrescriben a los del original.

▼ Sintaxis

```
diccionario1.update(diccionario2)
```

1 Empieza a programar o a [crear código](#) con IA.

```
1 perfil_usuario = {  
2     "usuario": "dev_master",  
3     "email": "user@example.com"  
4 }  
5  
6 datos_adicionales = {  
7     "pais": "España",  
8     "email": "new_email@example.com" # Esta clave se actualizará  
9 }  
10  
11 print(f"Perfil original: {perfil_usuario}")  
12  
13
```

Perfil original: {'usuario': 'dev_master', 'email': '[user@example.com](#)'}

```
1 # Actualizamos el perfil con los nuevos datos  
2 perfil_usuario.update(datos_adicionales)  
3  
4 print(f"Perfil actualizado: {perfil_usuario}")
```

Perfil actualizado: {'usuario': 'dev_master', 'email': '[new_email@example.com](#)', 'pais': 'España'}

▼ Ejercicio Propuesto

- Tienes un diccionario `inventario_parcial = {"manzanas": 20, "naranjas": 15}`. Tienes otro diccionario con una nueva entrega: `nueva_entrega = {"naranjas": 30, "platanos": 25}`. Actualiza `inventario_parcial` con la `nueva_entrega` y muestra el resultado final.

▼ 3. Caso de Uso: Gestión de un Inventario Simple 🛒

Vas a gestionar el inventario de una pequeña tienda de fruta. Tu tarea es usar un diccionario para realizar un seguimiento de los productos y sus cantidades (en kg) a lo largo de un día de trabajo.

Sigue estos pasos:

1. Creación del Inventario:

- Crea un diccionario llamado `inventario`.
- Añade los siguientes productos con sus cantidades iniciales:
 - "manzanas": 50
 - "platanos": 75
 - "naranjas": 40
- Muestra el inventario inicial por pantalla

2. Elemento de lista

- Ha llegado un nuevo producto. Añade "peras" al inventario con una cantidad de 30 kg.
- Hemos recibido más manzanas. Actualiza la cantidad de "manzanas", añadiendo 25 kg a su valor actual.
- Muestra el inventario después de estas actualizaciones.

3. Registro de una Venta:

- Un cliente ha comprado 10 kg de "naranjas". Resta esta cantidad del stock.
- Muestra el inventario actualizado.

4. Producto Agotado:

- Los plátanos se han estropeado y hay que retirarlos. Elimina "platanos" del inventario.
- Muestra el inventario final.

5. Consulta de Stock:

- El encargado te pregunta si quedan uvas. Utiliza el método .get() para consultar el stock de "uvas", proporcionando 0 como valor por defecto si no se encuentran. Imprime el resultado de la consulta.







6. Informe Final:

- Al final del día, genera un pequeño informe. Imprime un mensaje que diga "Informe Final de Inventario:" y, a continuación, recorre los items del diccionario final para mostrar cada fruta y su cantidad disponible en una línea separada (ej: - Hay 55 kg de manzanas).

4 Supongamos que un diccionario contiene la siguiente información sobre 5 empleados:

```
emp = {
    'A101': {'name': 'Ashish', 'age': 30, 'salary': 21000},
    'B102': {'name': 'Dinesh', 'age': 25, 'salary': 12200},
    'A103': {'name': 'Ramesh', 'age': 28, 'salary': 11000},
    'D104': {'name': 'Akheel', 'age': 30, 'salary': 18000},
    'A105': {'name': 'Akaash', 'age': 32, 'salary': 20000}
}
```

Escribe un programa en Python para crear:

-  Un diccionario con todos los códigos y sus respectivos valores, **solo si el código empieza con la letra 'A'**.
-  Un diccionario con todos los **códigos y nombres**.
-  Un diccionario con todos los **códigos y edades**.
-  Un diccionario con todos los **códigos y edades, solo si la edad es mayor que 30**.
-  Un diccionario con todos los **códigos y nombres, solo si el nombre comienza con la letra 'A'**.
-  Un diccionario con todos los **códigos y salarios, solo si el salario está entre 13,000 y 20,000 inclusive**.

✓ Salida esperada (Output)

```
{'A101': {'name': 'Ashish', 'age': 30, 'salary': 21000}, 'A103': {'name': 'Ramesh', 'age': 28, 'salary': 11000}}
```

```
{'A101': 'Ashish', 'B102': 'Dinesh', 'A103': 'Ramesh', 'D104': 'Akheel'}
```

```
{'A101': 30, 'B102': 25, 'A103': 28, 'D104': 30}
```

```
{}
```

```
{'A101': 'Ashish', 'D104': 'Akheel'}
```

```
{'D104': 18000}
```

5. Crear un diccionario con nombres de estudiantes y calificaciones

Cree un diccionario que contenga los nombres de los estudiantes y las calificaciones obtenidas por ellos en tres materias. Escriba un programa para reemplazar las calificaciones de tres materias con el total de tres materias y las calificaciones promedio. Informe también quién es el mejor de la clase.

6. Crear un diccionario de 10 nombres de usuario y contraseñas

El programa recibe el nombre de usuario y la contraseña por el teclado y luego los busca en el diccionario. Imprima el mensaje apropiado en la pantalla en función de si se encuentra una coincidencia o no.

7. Eliminar vocales de las claves de un diccionario

Dado un diccionario cuyas claves son cadenas de texto (strings), escribe un programa en Python que cree un nuevo diccionario

- eliminando todas las vocales (a, e, i, o, u) de dichas claves, manteniendo sus valores originales.

```
datos = {  
    'sol': 3,  
    'espacio': 7,  
    'cohetes': 6,  
    'tierra': 6  
}
```

✓ Salida esperada

```
{'sl': 3, 'spc': 7, 'cht': 6, 'trr': 6}
```

1 Empieza a programar o a [crear código](#) con IA.

- ## 8. Escribir un programa que cree un diccionario vacío y lo vaya llenado con información sobre una persona (por ejemplo nombre, edad, sexo, teléfono, correo electrónico, etc.) que se le pida al usuario. Cada vez que se añada un nuevo dato debe imprimirse el contenido del diccionario.

1 Empieza a programar o a [crear código](#) con IA.

9. Clientes

Escribir un programa que permita gestionar la base de datos de clientes de una empresa. Los clientes se guardarán en un diccionario en el que la clave de cada cliente será su NIF, y el valor será otro diccionario con los datos del cliente (nombre, dirección, teléfono, correo, preferente), donde preferente tendrá el valor True si se trata de un cliente preferente. El programa debe preguntar al usuario por una opción del siguiente menú: (1) Añadir cliente, (2) Eliminar cliente, (3) Mostrar cliente, (4) Listar todos los clientes, (5) Listar clientes preferentes, (6) Terminar. En función de la opción elegida el programa tendrá que hacer lo siguiente:

1. Preguntar los datos del cliente, crear un diccionario con los datos y añadirlo a la base de datos.
2. Preguntar por el NIF del cliente y eliminar sus datos de la base de datos.
3. Preguntar por el NIF del cliente y mostrar sus datos.
4. Mostrar lista de todos los clientes de la base datos con su NIF y nombre.
5. Mostrar la lista de clientes preferentes de la base de datos con su NIF y nombre.
6. Terminar el programa.

1 Empieza a programar o a [crear código](#) con IA.