



Practica 09 - Diccionarios



Instrucciones

Entregar los ejercicios en uno o varios [archivos.py](#) o en un notebook de Jupyter ([.ipynb](#)).

Esta tarea contiene 13 ejercicios a partir de los cuales debes entregar 7 ejercicios mas el ejercicio obligatorio (#13).

Total a entregar 8 ejercicios en un notebook o en un [archivo.py](#) o varios archivos.py



1 - 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.



2 - Crear un diccionario que almacene los siguientes datos:

Interface	IP	Address status
eth0	1.1.1.1	up
eth1	2.2.2.2	up
wlan0	3.3.3.3	down
wlan1	4.4.4.4	up

Escriba un programa para realizar las siguientes operaciones:

- Encontrar el estado de una interfaz determinada.
- Encontrar la interfaz y la IP de todas las interfaces que están activas.
- Encontrar el número total de interfaces.
- Añadir dos nuevas entradas al diccionario.

3 - 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.

4 - Crear diccionario con claves del 1 al 15 y valores al cuadrado

Escriba un programa en Python que imprima un diccionario donde las claves sean números entre 1 y 15 (ambos incluidos) y los valores sean los cuadrados de sus claves respectivas.

5 - Sumar todos los valores de un diccionario

Escriba un programa Python que imprima la suma todos los valores de un diccionario.

Puede utilizar números enteros para los pares clave:valor.

6 - Multiplicar todos los valores de un diccionario

Escriba un programa en Python que multiplique todos los valores de un diccionario. Puede utilizar números enteros para los pares clave:valor.

7 - Contar la cantidad de valores por clave

Escriba un programa Python que cuente la cantidad de valores (values) que contiene cada clave (key) del diccionario. El valor (o valores) de cada clave están guardados en una lista.

Ejemplo:

```
dict = {
    'clave_1': [val1, val2],
    'clave_2': [val3, val4],
    'clave_n': [valn, valn+1]
}
```

8 - Ordenar valores alfabéticamente en listas dentro de un diccionario

Escriba un programa en Python que ordene los elementos de una lista (contenida como valores en un diccionario) alfabéticamente.

Ejemplo del diccionario:

```
profes = {
    'Math': ['Julia', 'Miguel', 'Francisco', 'Eduardo', 'Elizabeth'],
    'Physics': ['Antonio', 'Adela', 'Maria'],
    'Chemistry': ['Rosa', 'Adela']
}
```

9 - Combinar dos diccionarios y sumar valores por clave

Escriba un programa en Python que combine dos diccionarios en uno solo sumando los valores de aquellas claves que estén repetidas en dicho diccionario.

Datos de muestra:

```
d1 = {'a': 100, 'b': 200, 'c': 300}  
d2 = {'a': 300, 'b': 200, 'd': 400}
```

Salida esperada:

```
Counter({'a': 400, 'b': 400, 'd': 400, 'c': 300})
```



10 — 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.



Instrucciones adicionales:

- Utiliza una **comprensión de diccionario** (`dict comprehension`).
- Dentro de esta comprensión, usa una **comprensión de lista o generador** para construir las nuevas claves sin vocales.
- Recuerda que las vocales pueden estar en **minúsculas o mayúsculas** (opcional si deseas hacer la solución más robusta).



Ejemplo de entrada

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



Salida esperada

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

11 - 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}  
}
```

Usando comprensiones de diccionarios ([dictionary comprehensions](#)), 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}
```

Consejos

- Observa que los datos están organizados como un **diccionario anidado**, es decir, cada valor del diccionario `emp` también es otro diccionario.
- Para acceder al nombre `'Ashish'` dentro del diccionario, usamos la siguiente sintaxis:

```
emp['A101']['name']
```

- Para acceder a la edad `32`, usamos esta otra sintaxis:

```
emp['A105']['age']
```

12 - Transformación de Diccionario

¿Cómo convertir el siguiente diccionario:

```
d = {'AMOL': 20, 'ANIL': 12, 'SUNIL': 13, 'RAMESH': 10}
```

en este nuevo diccionario?

```
{'Amol': 400, 'Anil': 144, 'Sunil': 169, 'Ramesh': 100}
```

Objetivo

- Transformar las claves para que:
 - Solo la primera letra esté en mayúscula y el resto en minúscula.
- Elevar al cuadrado los valores asociados a cada clave.

13 - Caso de uso : Sistema de Gestión de Inventario para un Mercado Local



■ Contexto del Problema

Un mercado local está intentando modernizar su sistema de control de inventario. Actualmente registran los productos a mano, y necesitan un sistema digital sencillo que permita organizar su información correctamente.

Cada producto debe almacenar:

- Su **nombre**
- Su **precio**
- Su **stock disponible**
- Los **códigos de lote**, que deben ser únicos (\rightarrow sets)
- Su **origen geográfico**, el cual no cambia (\rightarrow tupla)
- La categoría a la que pertenece (frutas, verduras, lácteos, etc.)

El mercado también maneja varias categorías, cada una con una lista propia de productos. Además, periódicamente combinan inventarios de diferentes sucursales y deben poder iterarlos, modificarlos y filtrarlos fácilmente.

Se pide:

Construir y manipular un sistema de inventario utilizando:

- **Diccionarios**
- **Listas**
- **Conjuntos (sets)**
- **Tuplas**

El objetivo NO es que escribas el código final, sino que sigas paso a paso el caso guiado y resuelvas cada apartado aplicando lo aprendido sobre diccionarios en la clase.

Pasos

1 Crear la estructura principal del inventario

Define un diccionario llamado `inventario` donde:

- Las **claves** sean categorías de productos.
- Los **valores** sean **listas vacías**, que luego contendrán diccionarios con los productos.

2 Registrar productos usando diccionarios, listas, sets y tuplas

Cada producto debe ser un diccionario que contenga:

- `'nombre'` : string
- `'precio'` : valor numérico
- `'stock'` : valor entero
- `'lotes'` : un **set** con códigos únicos
- `'origen'` : una **tupla** con `(país, región)`

Agrega al menos **3 productos por categoría**, insertándolos en las listas del inventario.

3 Acceder y consultar datos del inventario

Realiza las siguientes acciones:

- Accede al primer producto de alguna categoría.
- Obtén la ciudad/ubicación de origen desde la tupla.
- Usa `.get()` para acceder a una clave que probablemente no exista.
- Verifica si `'precio'` es una clave válida dentro del diccionario de un producto.

4 Modificar información del inventario

Debes:

- Actualizar el precio de un producto.
- Incrementar su stock.
- Añadir un nuevo lote único al set de lotes.

- Intentar modificar directamente la tupla `origen` (y reflexionar por qué no se puede).
- Crear una **nueva tupla corregida** y asignarla a `origen`.

5 Eliminar elementos del inventario

Realiza:

- Eliminar un producto de la lista usando su índice.
- Eliminar un código de lote con `.pop()` desde el set.
- Eliminar toda una categoría usando `del` o `.clear()`.

6 Combinar dos inventarios usando diccionarios

Crea un segundo diccionario `inventario_extra` con nuevas categorías.

Debes:

- Combinar ambos diccionarios usando `*dicc1, **dicc2`.
- Mezclarlos también usando `dicc1 | dicc2` (si tu versión lo permite).

En esta parte no debes modificar el inventario original.

7 Iterar sobre el inventario

Realiza:

- Iteración sobre categorías (`keys()`).
- Iteración sobre listas de productos (`values()`).
- Iteración de pares categoría-productos (`items()`).
- Iteración sobre la tupla `origen` de cada producto.

8 Diccionario por comprensión

Construye un diccionario donde:

- **Clave** → nombre del producto
- **Valor** → longitud de ese nombre

Con condición:

- Solo incluir productos cuyo nombre **no empiece por vocal**.

9 Usar `dict.fromkeys()`

Crea un diccionario donde:

- Las claves sean las categorías actuales
- El valor sea `0` (representando nivel de alerta)

10 Comprobación de hashabilidad

- Intenta usar una **lista** como clave en un diccionario (debe fallar).
- Usa una **tupla** como clave (debe funcionar).