Diccionarios

Hemos visto que las listas son útiles cuando se quiere agrupar valores en una estructura y acceder a cada uno de ellos a través del un valor numérico, un índice.

Otro tipo de estructura que nos permite referirnos a un determinado valor a través de un nombre o clave es un diccionario. Muchas veces este tipo de estructura es más apropiado que una lista.

El nombre diccionario da una idea sobre el propósito de la estructura ya que uno puede realizar fácilmente una búsqueda a partir de una palabra específica (clave) para obtener su definición (valor).

Un ejemplo podría ser una agenda telefónica, que nos permita obtener el número de teléfono de una persona a partir de su nombre.

Veamos entonces el modo de crear diccionarios.

```
agenda = {'Marado':'1552123', 'JPFeinman':'1523443', 'Dolina':'4584129', 'Spasiuk':'65748', 'Fontanarrosa':'32456'}
```

El acceso a un valor se realiza a partir de su clave, por ejemplo:

```
print(agenda['Marado'])
print(agenda['Fontanarrosa'])
1552123
32456
```

Los diccionarios consisten en pares llamados ítems formados por claves y sus valores correspondientes. En este ejemplo, los nombres son las claves y los números de teléfono son los valores. Cada clave es separada de su valor por los dos puntos (:), los ítems son separados por comas, y toda la estructura es encerrada entre llaves. Un diccionario vacío, sin ítems, se escribe con solo dos llaves: {}.

Las claves, debido a que funcionan como índices, no pueden ser repetidas. Veamos las formas más comunes de iterar sobre un diccionario:

```
# Imprime claves
print("Claves")
print("======")
for nom in agenda:
print(nom)
print()
print("Valores")
print("=======")
# Imprime valores
for tel in agenda.values():
print(tel)
print()
```

```
print("Clave y valor")Introducción al Desarrollo de Software - Unidad 3 - Parte B
print("============")
# Imprime items: clave valor
for nom, tel in agenda.items():
print(nom,tel)
```

```
Claves
======
JPFeinman
Spasiuk
Marado
Dolina
Fontanarrosa
Valores
=======
1523443
65748
1552123
4584129
32456
Clave y valor
============
JPFeinman 1523443
Spasiuk 65748
Marado 1552123
Dolina 4584129
Fontanarrosa 32456
```

Al igual que las listas, los diccionarios son sumamente flexibles y pueden estar formados por otros diccionarios (o inclusive listas). Analicemos un breve ejemplo de un diccionario que está conformado del siguiente modo:

• Cuenta con tres ítems

• El valor de cada ítem es otro diccionario que a su vez contiene tres ítems con las claves titulo, fecha y autor

A continuación veamos la implementación de esta estructura, la impresión manual y mediante iteración:

```
referencia = { "libro1":{"titulo":"El tutorial de Python","fecha":"2013","autor":"Guido van Rossum"},
"libro2":{"titulo":"Aprenda a Pensar Como un Programador con Python","fecha":"2002", "autor":"Allen Downey"},
"libro3":{"titulo":"Inmersión en Python 3", "fecha":"2009", "autor":"Mark Pilgrim"}
}

# acceso a los valores de titulo de cada libro
print("Títulos")
```

```
print("=======")
print(referencia["libro1"]["titulo"])Introducción al Desarrollo de Software - Unidad 3 - Parte B
print(referencia["libro2"]["titulo"])
print(referencia["libro3"]["titulo"])
print()
# Mezcladito
for clave in referencia:
print(clave)
print("======")
for clave2, val in referencia[clave].items():
print(clave2, val, sep=": ")
print()
Títulos
=======
El tutorial de Python
Aprenda a Pensar Como un Programador con Python
Inmersión en Python 3
libro3
======
autor: Mark Pilgrim
titulo: Inmersión en Python 3
fecha: 2009
libro2
======
autor: Allen Downey
titulo: Aprenda a Pensar Como un Programador con Python
fecha: 2002
libro1
======
autor: Guido van Rossum
titulo: El tutorial de Python
fecha: 2013
```

1 Operaciones
• len(d) retorna el número de items (pares clave-valor) en d
• d[k] retorna el valor asociado con la clave k
• d[k] = v asocia el valor v con la clave k
• del d[k] elimina el item con clave k
• k in d evalúa si existe un item en d que tenga la clave k

Aunque las listas y los diccionarios comparten varias características en común, existen ciertas distinciones importantes:

• Tipos de claves: Las claves de los diccionarios no deben ser enteros (aunque pueden serlo). Deben ser tipos de datos inmutables (números flotantes, cadenas de caracteres o tuplas)

• Agregado automático: En un diccionario se crea un ítem automáticamente al asignar un valor a una clave inexistente, en una lista no se puede agregar un valor en un índice que esté fuera del rango.
• Contenido: La expresión k in d (d es un diccionario) evalúa por la existencia de una clave, no de un valor. Por otro lado, la expresión v in l (siendo l una lista), busca por un valor en vez de por un índice.

Métodos

A continuación se describen brevemente algunos de los métodos más utilizados:
• clear() Elimina todos los ítems
• copy() Retorna una copia superficial del diccionario
• get(key[, default]) Retorna el valor de la clave key si existe, sino el valor default. Si no se proporciona un valor default, entonces retorna None.
• items() Retorna el par de valores del ítem clave, valor.
• keys() Retorna las claves.
• pop(key[, default]) Si la clave key está presente en el diccionario la elimina y retorna su valor, sino
retorna default. Si no se proporciona un valor default y la clave no existe se produce un error (KeyError).
• popitem() Elimina y retorna un par (clave, valor) arbitrario.
• setdefault(key[, default]) Si la clave key está presente en el diccionario retorna su valor. Si no, inserta la clave con un valor de default y retorna default
• update([other]) Actualiza los ítems de un diccionario en otro. Es útil para concatenar diccionarios.
• values() Retorna los valores del diccionario.
Los diccionarios pueden ser comparados por su igualdad si y solo si tienen los mismos ítems. Otras comparaciones (<, <=, >=, >) no son permitidas.