</ Ayudantía 9:
Diccionarios
/>

Rolando Rojas

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</ Temas a tratar

{01}
Introducción a Diccionarios

{02}
Uso de diccionarios

{03}
Consideraciones importantes

{04}
Recorriendo diccionarios

{05}
Ejercicios
Preguntas
Tarea

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</ Introducción a Diccionarios

Un diccionario es una estructura de datos no ordenada con pares clave-valor. Cada elemento en un diccionario tiene una clave única asociada a un valor. La clave funciona como un identificador que permite acceder al valor asociado.
En el caso de Python, un diccionario se define mediante llaves {} y consiste en una serie de pares clave-valor separados por comas:

```
diccionario_traduccion = {'hello': 'hola', 'goodbye': 'adiós', 'dog': 'perro'}
                          clave    valor   clave    valor   clave   valor
```

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</ Introducción a Diccionarios [2/2]

Es como una lista, pero en lugar de usar índices numéricos para acceder a los elementos, utilizas "claves" que pueden ser palabras o frases. Cada "clave" tiene asociado un "valor". Es como un diccionario de palabras en el que buscas la definición de una palabra (la clave) y obtienes la información relacionada (el valor).

Creación de diccionario:

```
numeros_diccionario = {'uno': 1, 'dos': 2, 'tres': 3, 'cuatro': 4}
```

Acceder a valor de diccionario:

```
print(numeros_diccionario['tres'])  # Salida: 3
```

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</ Uso de diccionarios

Creación de diccionario vacio:

```
mi_diccionario_vacio = {}
```

Agregar pares clave-valor:

```
mi_diccionario_vacio['nombre'] = 'John'
mi_diccionario_vacio['edad'] = 25
mi_diccionario_vacio['ciudad'] = 'Ejemploville'
```

```
{'nombre': 'John', 'edad': 25, 'ciudad': 'Ejemploville'}
```

Acceder a valores utilizando clave:

```
print(mi_diccionario_vacio['nombre'])  # Salida: 'John'
print(mi_diccionario_vacio['edad'])    # Salida: 25
print(mi_diccionario_vacio['ciudad'])  # Salida: 'Ejemploville'
```

Modificar un valor:

```
mi_diccionario_vacio['edad'] = 26
```

```
{'nombre': 'John', 'edad': 26, 'ciudad': 'Ejemploville'}
```

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</ Uso de diccionarios

```
{'nombre': 'John', 'edad': 25, 'ciudad': 'Ejemploville'}
```

Consultar sobre una llave:

```
'telefono' in mi_diccionario_vacio
```

```
False
```

```
'nombre' in mi_diccionario_vacio
```

```
True
```

Eliminar un par clave-valor:

```
del mi_diccionario['edad']
```

```
{'nombre': 'John', 'ciudad': 'Ejemploville'}
```

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</ Consideraciones importantes

Digamos que queremos hacer una copia de un diccionario, lo asignaremos a otra variable:

```
# Diccionario original
diccionario_original = {'nombre': 'John', 'edad': 25, 'ciudad': 'Ejemploville

# Asignar el diccionario a una nueva variable (paso por referencia)
diccionario_referencia = diccionario_original

# Modificar la variable de referencia
diccionario_referencia['edad'] = 26

# Imprimir ambos diccionarios
print("Diccionario original:", diccionario_original)
print("Diccionario por referencia:", diccionario_referencia)
```

Salida:

```
Diccionario original: {'nombre': 'John', 'edad': 26, 'ciudad': 'Ejemploville'}
Diccionario por referencia: {'nombre': 'John', 'edad': 26, 'ciudad': 'Ejemploville'}
```

¿Por qué si modificamos solo la variable "diccionario_referencia" también se modificó el original? ***Se pasó por referencia***

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</ Consideraciones importantes

¿Cómo copiar un diccionario entonces? Usar **dict()**

```
# Diccionario original
diccionario_original = {'nombre': 'John', 'edad': 25, 'ciudad': 'Ejemploville

# Copiar el diccionario utilizando el constructor dict()
diccionario_copia = dict(diccionario_original)

# Modificar la copia
diccionario_copia['edad'] = 26

# Imprimir ambos diccionarios
print("Diccionario original:", diccionario_original)
print("Diccionario copia:", diccionario_copia)
```

Salida:

```
Diccionario original: {'nombre': 'John',
'edad': 25, 'ciudad': 'Ejemploville'}
Diccionario copia: {'nombre': 'John',
'edad': 26, 'ciudad': 'Ejemploville'}
```

Ahora sí se modificó solamente el segundo diccionario.

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</ Recorriendo diccionarios

Ciclo for:

```
# Diccionario original
mi_diccionario = {'nombre': 'John', 'edad': 25, 'ciudad': 'Ejemploville'}

# Recorrer el diccionario e imprimir claves y valores
for clave in mi_diccionario:
    valor = mi_diccionario[clave]
    print("Clave:", clave, ", Valor:", valor)
```

```
Clave: nombre , Valor: John
Clave: edad , Valor: 25
Clave: ciudad , Valor: Ejemploville
```

El ciclo for recorre las claves del diccionario, y dentro del bucle, se accede a los valores asociados a esas claves para realizar alguna operación, en este caso, imprimir las claves y valores en cada iteración.

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</ Ejercicio 1

Jimmy McGill, un prestigioso abogado, requiere ayuda con el manejo de datos de sus clientes. Teniendo el diccionario clientes cuyas llaves son nombres y sus valores otros diccionarios. Estos diccionarios asociados a cada abogado contienen datos de:

- *Los juicios solicitados por mes, bajo la llave ’juicios’ que asocia una lista de listas con la forma[mes, cantidad].*
- *La edad del cliente*
- *Las empresas que ha defendido el abogado bajo la llave ’empresas’ que asocia una lista de strings*

```
clientes={ 'mike’ : { 'juicios’ : [ ['julio’ , 3], ['agosto’ , 1], ['octubre’ ,
4]],            ‘edad’ : 35,
'empresas’ : [ 'google’ , 'samsung’ , 'ciac’ ] } ,

'rachel’ : { 'juicios’ : [ [ 'enero’ , 3] , [ 'marzo’ , 4 ] , [ 'julio’ , 8 ] ],
‘edad’ : 70,
'empresas’ : [ 'google’ , 'codelco’ ] } ,

'harvey’ : { 'juicios’ : [ [ 'enero’ , 5 ] , [ 'febrero’ , 12] , [ 'julio’ , 24]
],              ‘edad’ : 18,
'empresas’ : [ 'mesa verde’ , 'samsung’ ] }    }
```

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</ Ejercicio 1

Se deben crear las siguientes funciones:
***juicios_por_mes(clientes)**: que reciba el diccionario clientes y retorne un diccionario que asocie el mes con la cantidad total de juicios realizados.*

```
>>juicios_por_mes(clientes)
>>{'julio': 35, 'marzo': 4, 'agosto': 1, 'enero': 8,
'febrero': 12, 'octubre': 4}
```

***total_juicios(clientes)** que reciba el nombre de un cliente y retorne un entero con la cantidad total de juicios en los que ha estado. Asuma que el diccionario clientes es una variable global del programa.*

```
>>total_juicios('harvey')
>>41
```

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</ Ejercicio 1: solución

```
def juicios_por_mes(clientes):
    juicios={}
    for llave in clientes:
        diccionario = clientes[llave]
        for mes,cantidad in diccionario['juicios']:
            if mes not in juicios:
                juicios[mes]=0
            juicios[mes]+=cantidad
    return juicios

print(juicios_por_mes(clientes))


def total_juicios(abogado):
    total=0
    for mes,cantidad in clientes[abogado]['juicios']:
        total+=cantidad
    return total
print(total_juicios('harvey'))
```

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

No hay tarea esta semana

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</ Ayudantía 9:
Diccionarios
/>

¿Dudas?
rolando.rojass@usm.cl

Rolando Rojas

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1