Estructuras de datos y funciones

Métodos con diccionarios

{desafío} latam_

Utilizar estructuras de datos apropiadas para la elaboración de un algoritmo que resuelve un problema acorde al lenguaje Python.

Codificar un programa utilizando funciones para la reutilización de código acorde al lenguaje Python.

- Unidad 1:
  Introducción a Python
- Unidad 2:
  Sentencias condicionales e iterativas
- Unidad 3:
  Estructuras de datos y funciones

Te encuentras aquí

{desafío} latam_

¿Qué aprenderás en esta sesión?

- *Aplica métodos para la manipulación de diccionarios*

{desafío}
latam_

¿Qué son los diccionarios?

{desafío} latam_

/* Métodos con Diccionarios */

Agregar un elemento

```
diccionario = {"llave 1": 5}
```

Para agregar un elemento a un diccionario, hace falta especificar una clave nueva, de forma que el nuevo valor ingresado sea identificado con dicha clave.

```
diccionario["llave 2"] = 9
print(diccionario) # {"llave 1": 5, "llave 2": 9}
```

Presta atención a la sintaxis, donde para definir un diccionario usamos llaves ({}), pero para acceder a sus elementos usamos corchetes [].

Cambiar un elemento

De forma similar a como agregamos un elemento a un diccionario, podemos actualizar el valor de uno. Para esto, simplemente tenemos que redefinir el valor asociado a la clave.

```
diccionario = {"llave 1": 5, "llave 2": 7}
diccionario["llave 2"] = 9
print(diccionario) # {"llave 1": 5, "llave 2": 9}
```

Eliminar elementos

Podemos eliminar una llave de un diccionario, junto a su valor, de dos formas: usando el método **pop** del diccionario o utilizando **del**. La principal diferencia entre ambas formas es que al utilizar **pop** obtendremos el valor del elemento eliminado.

```
diccionario = {"celular": 140000, "notebook": 489990, "tablet": 120000, "cargador": 12400}
del diccionario["celular"]
print(diccionario)
```

```
{'notebook': 489990, 'tablet': 120000, 'cargador': 12400}
```

Eliminar elementos

Para usar pop se usa la clave, no la posición.

```
eliminado = diccionario.pop("tablet")

print(eliminado)
print(diccionario)
```

```
120000
{'notebook': 489990, 'cargador': 12400}
```

Unir diccionarios

```
diccionario_a = {"nombre": "Alejandra", "apellido": "López", "edad": 33, "altura": 1.55}
diccionario_b = { "mascota":"miti", "ejercicio":"bicicleta"}

# Union de diccionario_a y diccionario_b
diccionario_a.update(diccionario_b)

# Notar que la unión queda en el primer diccionario
print(diccionario_a)
```

```
{'nombre': 'Alejandra', 'apellido': 'López', 'edad': 33, 'altura': 1.55, 'mascota': 'miti',
'ejercicio': 'bicicleta'}
```

¡Cuidado con las colisiones!

Cuando ambos diccionarios tienen una clave en común, el valor del segundo diccionario sobreescribe al del primero.

```
diccionario_a = {"nombre": "Alejandra", "apellido": "López", "edad": 33, "altura": 1.55}
diccionario_b = { "mascota":"miti", "ejercicio":"bicicleta", "altura": 155}

# Union de diccionario_a y diccionario_b
diccionario_a.update(diccionario_b)

# Se sobreescribió el valor de altura por el del diccionario_b
print(diccionario_a)
```

```
{'nombre': 'Alejandra', 'apellido': 'López', 'edad': 33, 'altura': 155, 'mascota': 'miti',
'ejercicio': 'bicicleta'}
```

¡Cuidado con las colisiones!

Cuando ambos diccionarios tienen una clave en común, el valor del segundo diccionario sobreescribe al del primero.

Por lo tanto, no es lo mismo hacer **`diccionario_a.update(diccionario_b)`** que **`diccionario_b.update(diccionario_a)`**

```
diccionario_a = {"nombre": "Alejandra", "apellido": "López", "edad": 33, "altura": 1.55}
diccionario_b = { "mascota":"miti", "ejercicio":"bicicleta", "altura": 155}
diccionario_b.update(diccionario_a)
print(diccionario_b)
```

```
{'mascota': 'miti', 'ejercicio': 'bicicleta', 'altura': 1.55, 'nombre': 'Alejandra',
'apellido': 'López', 'edad': 33}
```

Método keys()

Entrega una lista con todas las claves de un diccionario

```
computador = {'notebook': 489990, 'tablet': 120000, 'cargador': 12400}
```

```
computador.keys()
```

```
dict_keys(['notebook', 'tablet', 'cargador'])
```

Método values()

Entrega una lista con todos los valores de un diccionario

```
computador = {'notebook': 489990, 'tablet': 120000, 'cargador': 12400}
```

```
computador.values()
```

```
dict_values([489990, 120000, 12400])
```

Método items()

Entrega una lista con los pares clave-valor de un diccionario

```
computador = {'notebook': 489990, 'tablet': 120000, 'cargador': 12400}
```

```
computador.items()
```

```
dict_items([('notebook', 489990), ('tablet', 120000), ('cargador', 12400)])
```

Método get()

Entrega un mensaje alternativo en caso de no encontrar alguna clave

```
computador = {'notebook': 489990, 'tablet': 120000, 'cargador': 12400}
```

```
computador.get('iphone','No se encuentra el elemento solicitado')
```

```
'No se encuentra el elemento solicitado'
```

/* Convertir estructuras */

Convertir un diccionario en una lista

Para lograr esto, se debe utilizar la función items(). Cada par (clave, valor) será una tupla:

```
list({"k1": 5, "k2": 7}.items()) # [('k1', 5), ('k2', 7)]
```

Convertir una lista en un diccionario

Para invertir la transformación se utiliza la función dict.

```
dict([('k1', 5), ('k2', 7)]) # {"k1": 5, "k2": 7}
```

Análogamente existen las funciones **`tuple()`** y **`set()`** que permitirán transformar a tuplas y/o sets respectivamente.

¡Practiquemos!

¡Practiquemos!

Realicemos la siguiente actividad en grupos

1. Crea un diccionario
2. Agrega un elemento
3. Cambia un elemento
4. Elimina un elemento

{desafío} latam_

¿Cuál es la diferencia entre listas y diccionarios?

Próxima sesión...

Desafío evaluado.

{desafío}
latam_

{desafío}
latam_

Academia de
talentos digitales