

Módulo 1

LECCIÓN 4

Conjuntos y Diccionarios



Sets o Conjuntos

Los sets o conjuntos están definidos por llaves `{}` y, al igual que con las listas y tuplas, es posible almacenar varios datos en una sola variable. Estos datos pueden ser de diferentes tipos pero no admite valores repetidos (los permite pero no los muestra ni tiene en cuenta), los datos no están ordenados, y no pueden ser tipos de datos mutables, es decir, dentro de un set no pueden ser ni diccionarios ni listas y, no hay un índice al cual pueda acceder.

```
[ ] estudiantes_set = {'Juan', 'Pedro', 'Andrés', 'María', 'Johanna', 'Johanna'}

print(estudiantes_set)

#Para acceder a un elemento debemos hacer un ciclo o preguntar si el elemento existe

for x in estudiantes_set:
    print(x)

print("Pedro" in estudiantes_set)
print("Carlos" in estudiantes_set)
```



```
{'Johanna', 'Pedro', 'Juan', 'María', 'Andrés'}
Johanna
Pedro
Juan
María
Andrés
True
False
```

Para un mejor entendimiento de los conjuntos, hay algunos elementos de la teoría de conjuntos, matemáticas, que ayudan a comprender de mejor manera los métodos disponibles.



Ejercitación Teoría de conjuntos

- Qué ocurre con la unión de dos conjuntos?
- Y con la intersección?
- Qué tal la diferencia?

Como se menciona al inicio de la lección, para entender mejor su uso, es recomendable entender un poco de la teoría de conjuntos.

Ejercitación Conjuntos al cuadrado

Crear un nuevo conjunto a partir del conjunto $A = \{1, 2, 3, 4, 5\}$ cuyos valores sean el cuadrado de los elementos del conjunto A . Recomendación, utilizar set comprehension.

```
{s for s in [1, 2, 1, 0]}
```

Es una estructura de datos que almacena la información en un algo llamado par llave valor, el cual se compone de una llave o nombre de variable y un valor que es la información almacenada.

Desde Python 3.7 los diccionarios son ordenado y similar a los conjuntos, se definen utilizando llaves `{}`

Los diccionarios tienen una estructura más robusta y agrupada para almacenar los datos. Permite almacenar datos de cualquier tipo mezclados. Las llaves de los diccionarios son pueden ser repetidas.

También permite guardar en sus valores, otras estructuras como listas y diccionarios.

Aquí el listado de métodos disponible para los diccionarios



```
[ ] #Un estudiante puede ser almacenado como un diccionario

juan = {
    "nombre": "Juan",
    "apellido": "Doe",
    "edad" : 10,
    "curso": 5,
    "materias" : ['inglés', 'matemáticas']
}

print(juan)

#Para acceder a un valor, debemos hacerlo a través de las llaves
# Utilizando notación de corchetes
print(juan["nombre"])

#o utilizando método get
juan.get('edad')

#En caso de no conocer el diccionario, es posible obtener todas las llaves, sin imprimir el diccionario completo
print(juan.keys())

#también puede imprimir solo los valores
print(juan.values())

#y modificar valores

juan["materias"] = [{
    "asignatura" : "ingles",
    "nota": 4
}, {
    "asignatura" : "matemáticas",
    "nota": 4.5
}]
print(juan)

# También obtener los pares llave valor en tuplas
juan.items()
```

juan.view

```
{'nombre': 'Juan', 'apellido': 'Doe', 'edad': 10, 'curso': 5, 'materias': ['inglés', 'matemáticas']}
Juan
dict_keys(['nombre', 'apellido', 'edad', 'curso', 'materias'])
dict_values(['Juan', 'Doe', 10, 5, ['inglés', 'matemáticas']])
{'nombre': 'Juan', 'apellido': 'Doe', 'edad': 10, 'curso': 5, 'materias': [{'asignatura': 'ingles', 'nota': 4}, {'asignatura': 'matemáticas', 'nota': 4.5}]}
dict_items([('nombre', 'Juan'), ('apellido', 'Doe'), ('edad', 10), ('curso', 5), ('materias', [{'asignatura': 'ingles', 'nota': 4}, {'asignatura': 'matemáticas', 'nota': 4.5}]])
```