Lección 09: Conjuntos (Sets)

Índice

Anterior

Definición

Hay cuatro estructuras de datos compuestas en Python: *Listas*, *Tuplas*, *Conjuntos* y *Diccionarios*.

Los conjuntos o sets : - Son una colección de elementos. - Son mutables, se pueden añadir y quitar elementos. - No estan ordenados, ni indexados. - No permite valores duplicados. - Pueden estar vacíos. - Puede contener elementos de distintos tipos, es decir cualquier tipo de objeto.

> **Nota:** Se pueden asimilar a la definición matemática de conjunto.

Se pueden crear vacíos con el contructor set(), o con elementos utilizando el constructor set() o llaves {}.

> **Nota:** En Python las llaves {} se utilizan para conjuntos y diccionarios. Unas llaves sin elementos dentro creará un diccionario y no un conjunto.

```
# Creación de conjuntos

conjunto1 = set() # Esto es un conjunto vacio
conjunto2 = {} # ¿Conjunto vacio? Ojo: Esto crea un diccionario vacío porque las llaves tam

print(conjunto1)
print(conjunto2)

print(type(conjunto1))
print(type(conjunto2)) #Ojo, esto es un diccionario vacío

# Creación de conjuntos con valores iniciales

conjunto3 = {"elemento1", "elemento2", "elemento3", 4}  # No pueden contener elementos repe
conjunto4 = set(["queso", "plátano", " ", 35, 44, 2.2, None, "uno"," "]) # ¿Qué ocurren si s

print(conjunto3)
print(conjunto4)

print(type(conjunto3))
print(type(conjunto4)) # Si se utiliza el constructor set() para crear un conjunto a partir
```

Se puede utilizar la función built-in *len()* para saber el número de elementos de un conjunto.

```
# Número de elementos de un conjunto

print("Conjunto:", conjunto3)
print(f"Número de elementos: {len(conjunto3)}")

print("Tupla:", conjunto4)
print(f"Número de elementos: {len(conjunto4)}")
```

Accediendo a los elementos de un conjunto

Los elementos de un conjunto no están ordenados, ni indexados. Para acceder a los elementos es necesario utilizar bucles.

Comprobando si un elemento está en un conjunto

- Se puede utilizar el operador *in*.

```
# Comprobando si un elemento está en un conjunto

globo_en_conjunto = "globo" in conjunto4
print(globo_en_conjunto)

none_en_conjunto = None in conjunto4
print(none_en_conjunto)

num_en_conjunto = 35 in conjunto4
print(num_en_conjunto)

cabra_en_conjunto = "cabra" in conjunto4
print(cabra_en_conjunto)
```

Añadiendo y eliminando elementos de un conjunto

- *add()* Añade un elemento al conjunto.
- *update()* Añade varios elementos a un conjunto, los elementos de una lista.
- *remove()* Elimina un elemento del conjunto. Si el elemento no existe dará un error.
- *discard()* Elimina un elemento de un conjunto, si existe. No dará error si el elemento no existe.
- *pop()* elimina un elemento aleatorio de un conjunto y lo devuelve.
- *clear()* elimina todos los elementos de un conjunto

```
# Añadiendo elementos a un conjunto

frutas = {"plátano", "tomate", "melón", "sandía", "naranja"}
print(frutas)
print(f'Número de elementos: {len(frutas)}')
```

```
frutas.add("mandarina")
print(frutas)
print(f'Número de elementos: {len(frutas)}')
frutas.update(["fresa","melocotón","ciruela"])
print(frutas)
print(f'Número de elementos: {len(frutas)}')

# Eliminando elementos de un conjunto

frutas.remove("platano") # Dará error, el elemento platano no existe en la lista

frutas.discard("granada") # Aunque el elemento no existe, no da error
print(frutas)
print(f'Número de elementos: {len(frutas)}')
frutas.remove("plátano")
print(frutas)
print(f'Número de elementos: {len(frutas)}')
eliminado = frutas.pop()
print(f'El elemento eliminado es: {eliminado}')
frutas.clear()
print(frutas)
print(f'Número de elementos: {len(frutas)}')
```

Eliminando un conjunto

- *del* elimina un conjunto, destruye el objeto.

```
# Eliminando un conjunto

del conjunto4
print(onjunto4)
```

Uniendo conjuntos

Se pueden unir conjuntos utilizando los métodos *union* o *update*.

- *union* devuelve un nuevo conjunto con la unión.
- *update* modifica el conjunto sobre el que se ejecuta el método (se utiliza también para añadir elementos).

```
## Uniendo conjuntos

conjunto3 = {"elemento1", "elemento2", "elemento3", 4}  # No pueden contener elementos repe
conjunto4 = set(["queso", "plátano", " ", 35, 44, 2.2, None, "uno"," "]) # ¿Qué ocurren si s

conjunto5 = conjunto3.union(conjunto4)
print(conjunto3)
print(conjunto4)
```

```
print(conjunto5)

conjunto3.update(conjunto4)
print(conjunto3)
```

Intersección entre conjuntos (Encontrando elementos comunes)

- *intesection()* Devuelve un conjunto con los elementos comunes.

```
## Encontrando elementos comunes

print(conjunto4)
print(conjunto5)
print(conjunto4.intersection(conjunto5))
```

Siguiente