Conjuntos

Generalidades

Los conjuntos son colecciones **desordenadas** que **no aceptan valores repetidos**. Los conjuntos permiten cambiar su contenido pero dado que no poseen un orden, **no aceptan operaciones indexadas**.

Los conjuntos son ampliamente utilizados en lógica y matemática, y desde el lenguaje podemos sacar provecho de sus propiedades para crear código más eficiente y legible en menos tiempo.

En Python, puedes crear conjuntos utilizando llaves {} o la función set().

```
conjunto1 = {1, 2, 3}
print (conjunto1)
print (type(conjunto1))
```

```
     {1, 2, 3}
     <class 'set'>
```

```
conjunto2 = set({4, 5, 6})
print (conjunto2)
print (type(conjunto2))
```

Un conjunto puede ser creado por un range

```
mi_conjunto = set (range (1,9))
print (mi_conjunto)
```

```
     {1, 2, 3, 4, 5, 6, 7, 8}
```

Un conjunto no puede incluir objetos mutables como listas, diccionarios, e incluso otros conjuntos.

```
mi_conjunto = {2,5,6,2,3,[4,2,6]}
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-3-55bfe1395a8a> in <cell line: 1>()
----> 1 mi_conjunto = {2,5,6,2,3,[4,2,6]}

TypeError: unhashable type: 'list'
```

Next steps: Explain error

En los conjuntos el orden de inserción NO es necesariamente el orden de almacenamiento

```
mi_conjunto = {'P', 'H', 'A', 'Z'}
print (mi_conjunto)
```

```
{'H', 'Z', 'A', 'P'}
```

Un conjunto en python no puede ser indexado, dado que no posee un orden

```
conj = {4,5,8,2}
print (conj.index(4))
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-5-61612cbf5700> in <cell line: 2>()
      1 conj = {4,5,8,2}
----> 2 print (conj.index(4))

AttributeError: 'set' object has no attribute 'index'
```

Next steps: Explain error

Tampoco puedo acceder a un elemento del connjunto por su indice

```
lista1={6,9,1,2,3,5}
print(lista1[0])
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-6-f3516a39a7ba> in <cell line: 2>()
      1 lista1={6,9,1,2,3,5}
----> 2 print(lista1[0])

TypeError: 'set' object is not subscriptable
```

Next steps: Explain error

Operaciones básicas con conjuntos

Adicción de Elementos

Para adicionar elementos al conjunto utilizamos el método **add()**

```
mi_conjunto = set (range (1,9))
print (mi_conjunto)
mi_conjunto.add(30)
print (mi_conjunto)
```

```
{1, 2, 3, 4, 5, 6, 7, 8}
{1, 2, 3, 4, 5, 6, 7, 8, 30}
```

Si debemos agregar varios elemento a un conjunto lo realizamos con el método "update()"

```
conjunto1 = set(range(1, 9))
print("Conjunto inicial:", conjunto1)

conjunto1.update([30, 10, 23, 55, 2, 11])

print("Conjunto con elementos agregados:", conjunto1)
```

```
Conjunto inicial: {1, 2, 3, 4, 5, 6, 7, 8}
Conjunto con elementos agregados: {1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 55, 23, 30}
```

En la creación de conjuntos podemos observar que NO permite adicionar elementos repetido.

```
mi_conjunto = set()
mi_conjunto.add(8)
mi_conjunto.add(2)
mi_conjunto.add(8)
mi_conjunto.add(7)
mi_conjunto.add(3)
mi_conjunto.add(8)
mi_conjunto.add(8)
print (len(mi_conjunto))
print (mi_conjunto)
```

```
4
{8, 2, 3, 7}
```

Comprobamos si un elemento se encuentra o no dentro de un conjunto con la palabra reservada **in**

```
mi_conjunto = {3,4,6,8,2}
print (mi_conjunto)
num = int(input("Cual número desea saber si está en el conjunto:"))
print (f"Está el nummero {num} en el conjunto {mi_conjunto}?: {num in mi_conjunto}")
```

```
{2, 3, 4, 6, 8}
Cual número desea saber si está en el conjunto:3
Está el nummero 3 en el conjunto {2, 3, 4, 6, 8}?: True
```

Eliminación de elementos en un conjunto

Para eliminar elementos de un conjunto, se utilizan los métodos remove() o discard(). La diferencia principal es que remove() generará un error si el elemento no está presente, mientras que discard() no.

discard() quita un elemento del conjunto

```
c = {2,5,7,9,3,1}
print(c)
c.discard (5)
print(c)
c.discard (4)
print(c)
```

```
{1, 2, 3, 5, 7, 9}
{1, 2, 3, 7, 9}
{1, 2, 3, 7, 9}
```

```
c = {2,5,7,9,3,1}
print(c)
c.remove (7)
print(c)
c.remove (6)
print(c)
```

```
{1, 2, 3, 5, 7, 9}
{1, 2, 3, 5, 9}
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-12-9aa2231e2cea> in <cell line: 5>()
      3 c.remove (7)
      4 print(c)
----> 5 c.remove (6)
      6 print(c)

KeyError: 6
```

Next steps: Explain error

Métodos en Conjuntos

El método clear() elimina todos los elementos del conjunto, dejándolo vacío.

```
conjunto = {1, 2, 3}
conjunto.clear()
print(conjunto)
```

```
set()
```

copy(): Crea una copia del conjunto.

```
conjunto_original = {1, 2, 3}
print("Conjunto original ",conjunto_original)
copia_conjunto = conjunto_original.copy()
print("Copia 1del conjunto:", copia_conjunto)
```

```
Conjunto original  {1, 2, 3}
Copia 1del conjunto: {1, 2, 3}
```

El método pop() elimina y devuelve un elemento aleatorio del conjunto.

```
conjunto = {"m", "b", "f", "i", "v", "n", "p"}

elemento = conjunto.pop()
print(elemento)  # Un elemento aleatorio
print(conjunto)
```

```
b
{'p', 'm', 'i', 'v', 'n', 'f'}
```

Operaciones con conjuntos

Los conjuntos en Python son una estructura de datos muy útil que nos permite almacenar colecciones de elementos únicos. Una de las características más poderosas de los conjuntos son las operaciones que podemos realizar con ellos, las cuales nos permiten combinar, comparar y manipular conjuntos de manera eficiente.

union() Une un conjunto a otro y devuelve el resultado en un nuevo conjunto:

```
c1 = {2,5,7}
c2 = {2,5,3}
c3 = c1.union(c2)
print (f"c1 unido con el c2 = {c3}")
```

```
     c1 unido con el c2 = {2, 3, 5, 7}
```

intersection() Devuelve un conjunto con los elementos comunes en dos conjuntos:

```
c1 = {2,7,5}
c2 = {2,5,3}
print (f"Los elementos comunes son = {c1.intersection(c2)}")
```

```
     Los elementos comunes son = {2, 5}
```

difference() Encuentra los elementos no comunes entre dos conjuntos:

```
c1 = {2,5,7}
c2 = {2,5,3}
print (f"Los elementos que solo estan en c1 son = {c1.difference(c2)}")
print (f"Los elementos que solo estan en c2 son = {c2.difference(c1)}")
```

```
     Los elementos que solo estan en c1 son = {7}
     Los elementos que solo estan en c2 son = {3}
```

symmetric_difference() Devuelve los elementos simétricamente diferentes entre dos conjuntos, es decir, todos los elementos que no concuerdan entre los dos conjuntos:

```
c1 = {2,5,7}
c2 = {2,5,3}
print (f"Los elementos diferentes en los dos conjuntos son = {c1.symmetric_difference(c2)}")
```

```
Los elementos diferentes en los dos conjuntos son = {3, 7}
```

issubset() Comprueba si el conjunto es subconjunto de otro conjunto, es decir, si sus ítems se encuentran todos dentro de otro:

```
c1 = {2,5,7}
c2 = {2,5}
print (f"c2 es subconjunto de c1 = {c2.issubset(c1)}")
```

```
c2 es subconjunto de c1 = False
```

issuperset() Comprueba si el conjunto es contenedor de otro subconjunto, es decir, si contiene todos los ítems de otro:

```
c1 = {2,5,7}
c2 = {2,5}
print (f"c1 contiene todos los elementos de c2? {c1.issuperset(c2)}")
```

```
c1 contiene todos los elementos de c2? True
```

isdisjoint() Comprueba si el conjunto es disjunto de otro conjunto, es decir, si no hay ningún elemento en común entre ellos:

```
c1 = {2,5,7}
c2 = {9,3,1}
print (f"c1 NO tiene elementos en comun con c2?  {c1.isdisjoint(c2)}")
c3 = {9,3,5}
print (f"c1 NO tiene elementos en comun con c3? {c2.isdisjoint(c3)}")
```

```
c1 NO tiene elementos en comun con c2?  True
c1 NO tiene elementos en comun con c3? False
```

Recorrido de conjuntos

Puedes recorrer un conjunto utilizando un bucle for.

```
colores = {"rojo", "verde", "azul", "amarillo", "rosa"}

for color in colores:
    print(color)
```

```
rojo
amarillo
rosa
```

```
azul
verde
```

O convertir el conjunto en una lista y luego recorrer esa lista utilizando el bucle deseado

```
colores = {"rojo", "verde", "azul", "amarillo", "rosa"}
indice = 0
color = list(colores)
while indice < len(colores):
    print(color[indice])
    indice += 1
```

```
rojo
amarillo
rosa
azul
verde
```

Si se recorre con while el conjunto nos mostraria todos los elementos en cada iteración, al no ser indexado no se verian los elementos individualmente.

```
colores = {"rojo", "verde", "azul", "amarillo", "rosa"}
indice = 0
cant = len(colores)
print (cant)
while indice<cant:
  print(colores)
  indice +=1
```

```
5
{'rojo', 'amarillo', 'rosa', 'azul', 'verde'}
{'rojo', 'amarillo', 'rosa', 'azul', 'verde'}
{'rojo', 'amarillo', 'rosa', 'azul', 'verde'}
{'rojo', 'amarillo', 'rosa', 'azul', 'verde'}
{'rojo', 'amarillo', 'rosa', 'azul', 'verde'}
```

Apropiación

1. Desarrolla un programa para gestionar clientes de una empresa. Utiliza un conjunto para almacenar los correos electrónicos de los clientes registrados. Permite al usuario agregar nuevos clientes, verificar si un cliente ya está registrado y eliminar clientes.

2. Crea un sistema para controlar el inventario de productos de una tienda. Utiliza un conjunto para almacenar los códigos de barras de los productos disponibles. Permite al usuario agregar

nuevos productos al inventario, verificar la disponibilidad de un producto y eliminar productos obsoletos.

3. Dos amigos se matricularon en la misma universidad, pero en diferentes carreras, ellos saben que existen materias denominadas institucionales y que deben ser vistas por los estudiantes de todas las carreras, se requiere un programa en Python que permita conocer la siguiente información:

   a. Identificar las materias en común que tendrán matriculadas en este semestre.

   b. El primer estudiante quiere saber cuáles materias tendrá que ver solo sin su amigo.

   c. El segundo estudiante quiere saber cuáles materias tendrán que ver el uno sin el otro.

   Para ello usted deberá construir de manera aleatoria dos conjuntos (cada uno con cuatro elementos) a partir de los elementos que están en la siguiente lista.

   Cada conjunto serán las materias que los amigos tendrá matriculadas, después de tener los dos conjuntos usted deberá hacer operaciones entre ellos para responder las preguntas de los estudiantes.

   Materias = ['Matematicas', 'Historia', 'Cátedra universitaria', 'Filosofía', 'Quimica','Fisica','Algebra', 'contabilidad', 'mercadeo', 'marketing', 'ambiental', 'geologia']

4. Se tiene las siguientes listas de documentos de estudiantes del CEAI inscritos a diferentes cursos. Los cursos son:

   ingles = [101, 102, 103, 104, 105, 106]

   matematicas = [104, 105, 107, 108, 109, 110]

   ciencias = [103, 104, 105, 110, 111, 112]

   El centro esta organizando un evento deportivo, sin embargo algunos estudiantes están inscritos en más de un curso, y no queremos contarlos varias veces al hacer la lista de asistencia. Queremos tener la siguiente informacion:

   a. Cantidad total de estudiantes con los que se cuentan para el evento deportivo.

   b. Saber cuales son los estudiantes que están inscritos en todos los cursos.

   c. Estudiantes que solo estan inscritos en un curso

   d. Poder comprobar la inscripción individualde un estudiante en particular en alguno de los cursos.

4. Aportes para actividades extracurriculares

Se les solicita a un grupo de estudiantes del CEAI proponer las actividades extracurriculares que mas les gustaria realizar, como el centro de formación tiene tres sedes, estas fueron las propuestas que se recogieron: lista_salomia = ['deportes', 'música', 'robótica', 'pintura','danza']

lista_norte = ['danza', 'fotografía', 'arte', 'música', 'robótica', 'arte', 'taller lectura', 'música', 'robótica','danza']

lista_tequedama = ['teatro', 'danza', 'ajedrez', 'robótica', 'deportes', 'taller lectura', 'club ciencias']

Ahora el equipo de bienestar necesitan cierta información a partir de las listas de datos y le han encargado a usted hacer un programa en python con conjuntos para ello:

a. listar todas las actividades propuestas e indicar cuantas son.

b. listar cuales son las actividades que en todas las sedes fueron propuestas (En común).

c. Verificar cuales son las actividades que solo las quiere cada sede.

d. Cual sede propuso más actividades y cuantas fueron.