

**ESTRUCTURA DE DATOS FUNDAMENTALES Y ALGORITMOS**  
**GUÍA DE LABORATORIO 05**  
**CONJUNTOS (SETS)**

Asignatura	Datos del alumno	Fecha y Firma
Algoritmos y solución de problemas	Apellidos:	
	Nombre:	

**Instrucciones:**

Desarrollar las actividades que indica el docente en base a la guía de trabajo que se presenta.

**1. Objetivos:**

- 🔧 Escribir algoritmos y codificar haciendo uso de conjuntos.

**2. Equipos, herramientas o materiales**

- 🔧 Computador, Software: Python, Algoritmos

**3. Fundamento teórico****3.1. Conceptos Clave****3.2. Conjuntos**

Un conjunto es una colección de elementos. La definición matemática de un conjunto también se puede aplicar en Python. Un conjunto es una colección de elementos distintos, desordenados y no indexados. En Python, el conjunto se usa para almacenar elementos únicos, y es posible encontrar la unión, intersección, diferencia, diferencia simétrica, subconjunto, superconjunto y conjunto disjunto entre conjuntos.

**Creando un Conjunto**

Usamos la función incorporada set().

- 🔧 Creando un conjunto vacío

```
1 #sintaxis
2 st = set()
```

- 🔧 Creando un conjunto con elementos iniciales

```
1 #sintaxis
2 st = {'elemento1', 'elemento2', 'elemento3', 'elemento4'}
3
4 #sintaxis
5 frutas = {'plátano', 'naranja', 'mango', 'limón'}
```

**Obteniendo la longitud de un Conjunto**

Usamos el método len() para encontrar la longitud de un conjunto.

```
1 #sintaxis
2 st = {'elemento1', 'elemento2', 'elemento3', 'elemento4'}
3 len(st)
4
5 #Ejemplo:
6 frutas = {'plátano', 'naranja', 'mango', 'limón'}
7 len(frutas)
```



### Verificando un elemento

Para verificar si un elemento existe en una lista, usamos el operador de pertenencia in.


```
1 #sintaxis
2 st = {'elemento1', 'elemento2', 'elemento3', 'elemento4'}
3 print("¿El conjunto st contiene elemento3? ", 'elemento3' in st) # ¿El conjunto st contiene elemento3? Verdadero
4
5 #Ejemplo:
6 frutas = {'plátano', 'naranja', 'mango', 'limón'}
7 print('mango' in frutas) # Verdadero
```

### Añadiendo elementos a un Conjunto

Una vez creado un conjunto, no podemos cambiar ninguno de sus elementos, pero también podemos agregar elementos adicionales.


 Agregue un elemento usando add()

```
1 #sintaxis
2 st = {'elemento1', 'elemento2', 'elemento3', 'elemento4'}
3 st.add('elemento5')
4
5 #Ejemplo:
6 frutas = {'plátano', 'naranja', 'mango', 'limón'}
7 frutas.add('lima')
```


 Agregar múltiples elementos usando update() El método update() permite agregar múltiples elementos a un conjunto. El método update() toma una lista como argumento.

```
1 #sintaxis
2 st = {'elemento1', 'elemento2', 'elemento3', 'elemento4'}
3 st.update(['elemento5', 'elemento6', 'elemento7'])
4
5 #Ejemplo:
6 frutas = {'plátano', 'naranja', 'mango', 'limón'}
7 verduras = ('tomate', 'patata', 'col', 'cebolla', 'zanahoria')
8 frutas.update(verduras)
```

### Eliminando elementos de un Conjunto

 Podemos eliminar un elemento de un conjunto usando el método remove(). Si el elemento no se encuentra, el método remove() provocará errores, por lo que es bueno verificar si el elemento existe en el conjunto dado. Sin embargo, el método discard() no provoca ningún error.

```
1 #sintaxis
2 st = {'elemento1', 'elemento2', 'elemento3', 'elemento4'}
3 st.remove('elemento2')
```

 El método pop() elimina un elemento aleatorio de una lista y devuelve el elemento eliminado.

```
1 frutas = {'plátano', 'naranja', 'mango', 'limón'}
2 frutas.pop() # elimina un elemento aleatorio del conjunto
3
4 #Si estamos interesados en el elemento eliminado.
5
6 frutas = {'plátano', 'naranja', 'mango', 'limón'}
7 elemento_eliminado = frutas.pop()
8 print(elemento_eliminado)
```

### Limpiando elementos en un conjunto

Si queremos limpiar o vaciar el conjunto, usamos el método clear().



```
1 #sintaxis
2 st = {'elemento1', 'elemento2', 'elemento3', 'elemento4'}
3 st.clear()
4
5 #Ejemplo:
6 frutas = {'plátano', 'naranja', 'mango', 'limón'}
7 frutas.clear()
8 print(frutas) # set()
```

### **Borrando un Conjunto**

Si queremos eliminar el conjunto en sí, usamos el operador del.

```
1 #sintaxis
2 st = {'elemento1', 'elemento2', 'elemento3', 'elemento4'}
3 del st
4
5 #Ejemplo:
6 frutas = {'plátano', 'naranja', 'mango', 'limón'}
7 del frutas
```

### **Convirtiendo una Lista a Conjunto**

Podemos convertir una lista en un conjunto y un conjunto en una lista. Convertir una lista a un conjunto elimina los duplicados y solo se reservarán los elementos únicos.


```
1 #sintaxis
2 lst = ['elemento1', 'elemento2', 'elemento3', 'elemento4', 'elemento1']
3 st = set(lst) # {'elemento2', 'elemento4', 'elemento1', 'elemento3'} - el orden es aleatorio, porque los conjuntos
4
5 #Ejemplo:
6 frutas = ['plátano', 'naranja', 'mango', 'limón', 'naranja', 'plátano']
7 frutas = set(frutas) # {'mango', 'limón', 'plátano', 'naranja'}
```

### **Uniendo Conjuntos (Union)**

Podemos unir dos conjuntos usando el método union() o update().

 **Unión** Este método devuelve un nuevo conjunto

```
1 #sintaxis
2 st1 = {'elemento1', 'elemento2', 'elemento3', 'elemento4'}
3 st2 = {'elemento5', 'elemento6', 'elemento7', 'elemento8'}
4 st3 = st1.union(st2)
5
6 #Ejemplo:
7 frutas = {'plátano', 'naranja', 'mango', 'limón'}
8 verduras = {'tomate', 'patata', 'col', 'cebolla', 'zanahoria'}
9 print(frutas.union(verduras)) # {'limón', 'zanahoria', 'tomate', 'plátano', 'mango', 'naranja', 'col', 'patata', 'cebolla'}
```

 **Update** Este método inserta un conjunto en un conjunto dado

```
1 #sintaxis
2 st1 = {'elemento1', 'elemento2', 'elemento3', 'elemento4'}
3 st2 = {'elemento5', 'elemento6', 'elemento7', 'elemento8'}
4 st1.update(st2) # el contenido de st2 se añade a st1
5
6 #Ejemplo:
7 frutas = {'plátano', 'naranja', 'mango', 'limón'}
8 verduras = {'tomate', 'patata', 'col', 'cebolla', 'zanahoria'}
9 frutas.update(verduras)
10 print(frutas) # {'limón', 'zanahoria', 'tomate', 'plátano', 'mango', 'naranja', 'col', 'patata', 'cebolla'}
```

### **Encontrando Elementos (Intersección)**

La intersección devuelve un conjunto de elementos que están en ambos conjuntos.



```
1 #sintaxis
2 st1 = {'elemento1', 'elemento2', 'elemento3', 'elemento4'}
3 st2 = {'elemento3', 'elemento2'}
4 st1.intersection(st2) # {'elemento3', 'elemento2'}
5
6 #Ejemplo:
7 numeros_enteros = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
8 numeros_pares = {0, 2, 4, 6, 8, 10}
9 numeros_enteros.intersection(numeros_pares) # {0, 2, 4, 6, 8, 10}
10
11 python = {'p', 'y', 't', 'h', 'o', 'n'}
12 dragon = {'d', 'r', 'a', 'g', 'o', 'n'}
13 python.intersection(dragon) # {'o', 'n'}
```

### Comprobando Subconjunto y Superconjunto

Un conjunto puede ser un subconjunto o un superconjunto de otros conjuntos:

🐡 Subconjunto: `issubset()`

🐡 Superconjunto: `issuperset()`

```
1 #sintaxis
2 st1 = {'elemento1', 'elemento2', 'elemento3', 'elemento4'}
3 st2 = {'elemento2', 'elemento3'}
4 st2.issubset(st1) # Verdadero
5 st1.issuperset(st2) # Verdadero
6
7 #Ejemplo:
8 numeros_enteros = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
9 numeros_pares = {0, 2, 4, 6, 8, 10}
10 numeros_enteros.issubset(numeros_pares) # Falso, porque es un superconjunto
11 numeros_enteros.issuperset(numeros_pares) # Verdadero
12
13 python = {'p', 'y', 't', 'h', 'o', 'n'}
14 dragon = {'d', 'r', 'a', 'g', 'o', 'n'}
15 python.issubset(dragon) # Falso
```

### Verificando la Diferencia entre dos Conjuntos

Devuelve la diferencia entre dos conjuntos.

```
1 #sintaxis
2 st1 = {'elemento1', 'elemento2', 'elemento3', 'elemento4'}
3 st2 = {'elemento2', 'elemento3'}
4 st2.difference(st1) # set()
5 st1.difference(st2) # {'elemento1', 'elemento4'} => st1\st2
6
7 #Ejemplo:
8 numeros_enteros = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
9 numeros_pares = {0, 2, 4, 6, 8, 10}
10 numeros_enteros.difference(numeros_pares) # {1, 3, 5, 7, 9}
11
12 python = {'p', 'y', 't', 'h', 'o', 'n'}
13 dragon = {'d', 'r', 'a', 'g', 'o', 'n'}
14 python.difference(dragon) # {'p', 'y', 't'} - el resultado no está ordenado (característica de los conjuntos)
15 dragon.difference(python) # {'d', 'r', 'a', 'g'}
```

### Encontrando la Diferencia Simétrica entre dos Conjuntos

Devuelve la diferencia simétrica entre dos conjuntos. Esto significa que devuelve un conjunto que contiene todos los elementos de ambos conjuntos, excepto los elementos que están presentes en ambos conjuntos, matemáticamente:

$$(A \setminus B) \cup (B \setminus A)$$



```
1 #sintaxis
2 st1 = {'elemento1', 'elemento2', 'elemento3', 'elemento4'}
3 st2 = {'elemento2', 'elemento3'}
4
5 #significa  $(A \setminus B) \cup (B \setminus A)$ 
6 st2.symmetric_difference(st1) # {'elemento1', 'elemento4'}
7
8 #Ejemplo:
9 numeros_enteros = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
10 algunos_numeros = {1, 2, 3, 4, 5}
11 numeros_enteros.symmetric_difference(algunos_numeros) # {0, 6, 7, 8, 9, 10}
12
13 python = {'p', 'y', 't', 'h', 'o', 'n'}
14 dragon = {'d', 'r', 'a', 'g', 'o', 'n'}
15 python.symmetric_difference(dragon) # {'r', 't', 'p', 'y', 'g', 'a', 'd', 'h'}
```

### Uniando Conjuntos Disjuntos

Si dos conjuntos no tienen un elemento o elementos comunes, los llamamos conjuntos disjuntos. Podemos comprobar si dos conjuntos son conjuntos o disjuntos usando el método `isdisjoint()`.

```
1 #sintaxis
2 st1 = {'elemento1', 'elemento2', 'elemento3', 'elemento4'}
3 st2 = {'elemento2', 'elemento3'}
4 st2.isdisjoint(st1) # Falso
5
6 #Ejemplo:
7 numeros_pares = {0, 2, 4, 6, 8}
8 numeros_impares = {1, 3, 5, 7, 9}
9 numeros_pares.isdisjoint(numeros_impares) # Verdadero, porque no hay elementos comunes
10
11 python = {'p', 'y', 't', 'h', 'o', 'n'}
12 dragon = {'d', 'r', 'a', 'g', 'o', 'n'}
13 python.isdisjoint(dragon) # Falso, hay elementos comunes {'o', 'n'}
```

## 4. Desarrollo y Actividades

### Ejercicio parte 01:

1. Escriba una función que reciba un conjunto de números y devuelva un conjunto con los números primos.
2. Escriba una función que reciba un conjunto de palabras y devuelva un conjunto con las palabras que comienzan con una letra determinada.
3. Escriba una función que reciba un conjunto de números y devuelva un conjunto con los números que son divisibles por un número determinado.
4. Escriba una función que reciba dos conjuntos de números y devuelva un conjunto con los números que están en ambos conjuntos.
5. Escriba una función que reciba dos conjuntos de números y devuelva un conjunto con los números que están en el primer conjunto pero no en el segundo.
6. Escriba una función que reciba dos conjuntos de números y devuelva un conjunto con los números que están en el segundo conjunto pero no en el primero.
7. Escriba una función que reciba un conjunto de palabras y devuelva un conjunto con las palabras que son anagramas.
8. Escriba una función que reciba un conjunto de palabras y devuelva un conjunto con las palabras que son palíndromos.
9. Escriba una función que reciba un conjunto de palabras y devuelva un conjunto con las palabras que tienen una longitud determinada.





10. Escriba una función que reciba un conjunto de palabras y devuelva un conjunto con las palabras que contienen una letra determinada.
11. Escriba una función que reciba un conjunto de números y devuelva un conjunto con los números que están ordenados de menor a mayor.
12. Escriba una función que reciba un conjunto de números y devuelva un conjunto con los números que están ordenados de mayor a menor.
13. Escriba una función que reciba un conjunto de números y devuelva un conjunto con los números que están duplicados.
14. Escriba una función que reciba un conjunto de números y devuelva un conjunto con los números que no están duplicados.
15. Escriba una función que reciba un conjunto de números y devuelva un conjunto con los números que son primos y están ordenados de menor a mayor.
16. Escriba una función que reciba un conjunto de palabras y devuelva un conjunto con las palabras que son palíndromos y están ordenadas de menor a mayor.
17. Escriba una función que reciba un conjunto de palabras y devuelva un conjunto con las palabras que tienen una longitud determinada y están ordenadas de menor a mayor.
18. Escriba una función que reciba un conjunto de palabras y devuelva un conjunto con las palabras que contienen una letra determinada y están ordenadas de mayor a menor.
19. Escriba una función que reciba un conjunto de números y devuelva un conjunto con los números que están ordenados de menor a mayor y que no están duplicados.
20. Escriba una función que reciba un conjunto de palabras y devuelva un conjunto con las palabras que son palíndromos, tienen una longitud determinada y están ordenadas de menor a mayor.

