

TEMA 3: Estructuras de datos

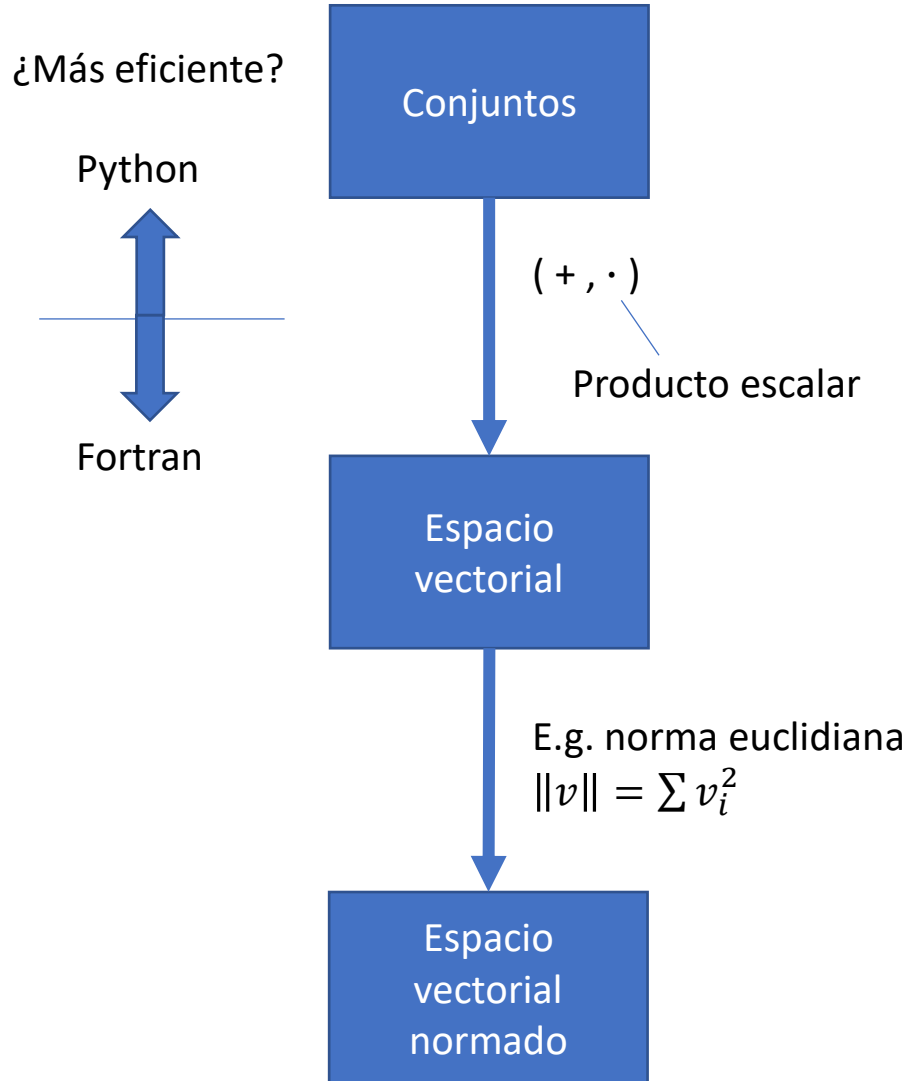
Dpto. de Matemática Aplicada a la Ingeniería Aeroespacial

Informática, 1^{er} semestre
2022 – 2023

Profesor: Juan Antonio Hernández Ramos (juanantonio.hernandez@upm.es)

Coordinador: Javier de Vicente Buendía (fj.devicente@upm.es)

Colaborador: Víctor Javier Llorente Lázaro (victorjavier.llorente@upm.es)



• ESTRUCTURAS DE DATOS

1. Conjunto: $S = \{1, 2, 3\}$
 - Tipo `<set>`
 - Objeto mutable
 - Colección no ordenada de elementos únicos
2. Lista: $L = [1, 2, 3]$
 - Tipo `<list>`
 - Objeto mutable
 - Colección ordenada de elementos no únicos
3. Tupla: $T = (1, 2, 3)$
 - Tipo `<tuple>`
 - Objeto inmutable
 - Colección ordenada de elementos no únicos
4. Diccionario: $D = \{\text{key1:value1, key2:(value2, value3)}\}$
 - Tipo `<dict>`
 - Objeto mutable
 - Colección no ordenada de Keys únicos y Values no únicos
5. Array (Tema 5)

• CONJUNTOS

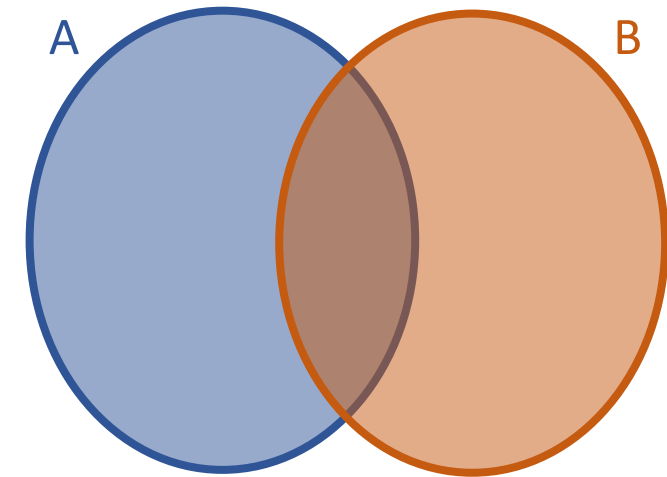
Los conjuntos de Python son análogos a los conjuntos matemáticos. Pueden estar formados por elementos distintos (heterogéneos)

```
s = {5, 4, 6, 8, 8, 1} # {1, 4, 5, 6, 8}
s = set()              # conjunto vacío
s = {5, 'hola', -625, 'ETSIA - UPM'}
```

Importante: Un elemento del <set> no puede ser ni un diccionario ni una lista
No tiene sentido intentar obtener un elemento usando un índice,
pero sí es posible iterar sobre un conjunto

```
# Conjunto
s = {5, 4, 6, 8, 8, 1}
# Acceder a los elementos del conjunto
for elemento in s:
    print(elemento)
```

1
4
5
6
8



Operaciones

- Diferencia: $A - B$
- Diferencia simétrica: $A \Delta B$
- Intersección: $A \cap B$
- Unión: $A \cup B$

- OPERACIONES CON CONJUNTOS

```
# Conjuntos
s1 = {1, 2, 3, 4, 5, 6}
s2 = {2, 4, 6, 8, 10}
s3 = {1, 2, 3}
s4 = {4, 5, 6}
# Operaciones sobre conjuntos
print("La longitud del conjunto s1 es ",len(s1))
print("La diferencia entre s1 y s2 es ",s1 - s2)
print("La diferencia simétrica entre s1 y s2 es ",s1 ^ s2)
print("La intersección entre s1, s2, s3, y s4 es ",s1 & s2 & s3 & s4)
print("La unión entre s1, s2, y s3 es ",s1 | s2 | s3)
print("¿s3 es subconjunto de s1? ",s3 < s1)
print("¿Esta 8 en s2 y no en s1? ",8 in s2 and 8 not in s1)
```

```
La longitud del conjunto s1 es 6
La diferencia entre s1 y s2 es {1, 3, 5}
La diferencia simétrica entre s1 y s2 es {1, 3, 5, 8, 10}
La intersección entre s1, s2, s3, y s4 es set()
La unión entre s1, s2, y s3 es {1, 2, 3, 4, 5, 6, 8, 10}
¿s3 es subconjunto de s1? True
¿Esta 8 en s2 y no en s1? True
```

• METODOS PARA CONJUNTOS

Añadir <elemento>

```
conjunto.add(elemento)
```

Añadir todos los elementos

```
conjunto.clear()
```

Copiar

```
conjunto.copy()
```

Eliminar <elemento>

```
conjunto.discard(elemento)
```

Eliminar <elemento>

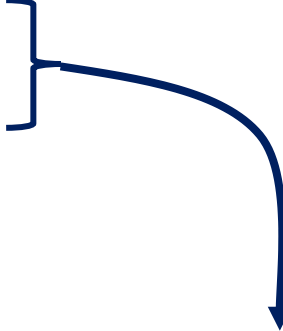
```
conjunto.remove(elemento)
```

Eliminar aleatoriamente

```
conjunto.pop()
```

Actualizar agregando un conjunto

```
lista.update(<set>)
```



`remove()` y `discard()` son métodos diferentes. El método `remove()` generará un error si el elemento especificado no existe, y el método `discard()` no lo hará.

• LISTAS

Las listas son heterogéneas, pueden estar formadas por elementos de distinto tipo; o incluso contener listas como elementos

```
lista = [5, 4, 6, 8]
lista = ['agua', 'fuego', 'tierra', 'aire']
lista = ['escuela de aeronáuticos', 54354]
lista = [['agua', 'fuego', 'tierra', 'aire'],
         ['escuela de aeronáuticos', 54354]]
```

¿Como acceder a los elementos de una lista?

variable[inicial:final:paso]

```
# Acceso a los elemento de una lista

pares = [0, 2, 4, 6, 8, 10, 12, 14, 16, 18]
print(pares[1      ])
print(pares[ :3     ])
print(pares[0:5: 2])
print(pares[9:0:-1])
```

```
2
[0, 2, 4]
[0, 4, 8]
[18, 16, 14, 12, 10, 8, 6, 4, 2]
```

- OPERACIONES CON LISTAS

```
# Concatenar
pares = [0, 2, 4, 6, 8]
impares = [1, 3, 5, 7, 9]
numeros = pares + impares
print(numeros)
```

```
[0, 2, 4, 6, 8, 1, 3, 5, 7, 9]
```

```
# Repetir
pares = [0, 2, 4, 6, 8]
pares = pares * 2 # pares *= 2
print(pares)
```

```
[0, 2, 4, 6, 8, 0, 2, 4, 6, 8]
```

```
# Comparación
pares = [0, 2, 4, 6, 8]
impares = [1, 3, 5, 7, 9]
print(pares > impares, pares >= impares)
print(pares < impares, pares <= impares)
print(pares == impares, pares != impares)
```

```
False False
True True
False True
```

```
# Bucles
pares = [0, 2, 4, 6, 8]
impares = [1, 3, 5, 7, 9]
numeros = pares + impares
for valor in numeros:
    if valor in pares:
        print(valor, " es par")
    else:
        print(valor, " es impar")
```

```
0 es par
2 es par
4 es par
6 es par
8 es par
1 es impar
3 es impar
5 es impar
7 es impar
9 es impar
```

• FUNCIONES PARA LISTAS

Longitud	<code>l = len(lista)</code>
Valor máximo	<code>m = max(lista)</code>
Valor mínimo	<code>m = min(lista)</code>
Eliminar elemento/lista	<code>del lista[indice] / del lista[:]</code>

• METODOS PARA LISTAS

Añadir† <elemento>	<code>lista.append(elemento)</code>
Añadir† elementos de un iterable	<code>lista.extend(<objeto>)</code>
Insertar <elemento> en el <indice> dado	<code>lista.insert(indice, elemento)</code>
Eliminar el elemento en el <indice> dado/último elemento	<code>lista.pop(indice) / lista.pop()</code>
Eliminar <elemento>	<code>lista.remove(elemento)</code>
Eliminar todos los elementos	<code>lista.clear()</code>
Contar veces que aparece <elemento>	<code>lista.count(elemento)</code>
Índice del <elemento>	<code>lista.index(elemento)</code>
Invertir orden de elementos	<code>lista.reverse()</code>
Ordenación ascendente/descendente	<code>lista.sort() / lista.sort(reverse=True)</code>

† al final de lista

- CREAR LISTAS

```
# Crear lista de cuadrados
N = 7
cuadrados = [0] * N
for i in range(N):
    cuadrados[i] = i ** 2
print(cuadrados)
```

Necesario inicializar la lista con ceros, por ejemplo, para poder asignar elementos a la lista

[0, 1, 4, 9, 16, 25, 36]

```
# Crear lista de cuadrados
N = 7
cuadrados = [i ** 2 for i in range(N)]
print(cuadrados)
```

[0, 1, 4, 9, 16, 25, 36]

- ENTRADA LISTA POR TECLADO

```
# Entrada de lista
datos = input("Dame los números: ").split()
media = 0.0
for numero in datos:
    media += float(numero)
media /= len(datos)
print("La media de los números es: ", media)
```

Los elementos de datos son de tipo <str>

Dame los números: 1 2 3 4
La media de los números es: 2.5

- COPIAR LISTAS

```
# Lista de cuadrados
cuadrados = [i ** 2 for i in range(4)]
# Copia profunda
cuadrados_cp = cuadrados
print(cuadrados, cuadrados_cp)
del cuadrados[2]
print(cuadrados, cuadrados_cp)
```

```
[0, 1, 4, 9] [0, 1, 4, 9]
[0, 1, 9] [0, 1, 9]
```

```
# Lista de cuadrados
cuadrados = [i ** 2 for i in range(4)]
# Copia superficial
cuadrados_cp = cuadrados[:]
print(cuadrados, cuadrados_cp)
del cuadrados[2]
print(cuadrados, cuadrados_cp)
```

```
[0, 1, 4, 9] [0, 1, 4, 9]
[0, 1, 9] [0, 1, 4, 9]
```

• TUPLAS

Las tuplas son como las listas pero **inmutables**: no se pueden modificar, borrar o añadir elementos

```
tupla = (5, 4, 6, 8)
tupla = ('agua', 'fuego', 'tierra', 'aire')
tupla = ('escuela de aeronáuticos', 54354)
tupla = ([ 'agua', 'fuego', 'tierra', 'aire'],
         ['escuela de aeronáuticos', 54354])
```

¿Cómo acceder a los elementos de una lista? `variable[inicial:final:paso]`

Operaciones, funciones, y métodos **permitidos** en tuplas:

```
+, *
in, not in
==, !=, >, >=, <, <=
len(tupla), max(tupla), min(tupla)
tupla.count(<obj>), tupla.index(<obj>[,index])
```

- **DICCIONARIOS**

Los diccionarios son una colección par de elementos, donde cada uno tiene una llave <key> y un(os) valor(es) <value>

```
diccionario = {'nombre': 'Felipe', 'edad': 54, 'DNI': 15264168}  
diccionario = {'dic1': {'a': 1, 'b': 2}, 'dic2': {'c': 3, 'd': 4}}
```

¿Como acceder a los elementos de una lista? `variable[<key>]`

```
# Diccionario  
d = {"nombre": "Felipe", "edad": 54, "DNI": 15264168}  
# Imprime los key del diccionario  
for llave in d:  
    print(llave)  
# Imprime los value del diccionario  
for llave in d:  
    print(d[llave])  
# Imprime los key y value del diccionario  
for llave, valor in d.items():  
    print(llave, valor)  
# Nueva llave  
d["direccion"] = ("C/ Falsa 123", "C/ lacalle 25")
```

nombre
edad
DNI

Felipe
54
15264168

nombre Felipe
edad 54
DNI 15264168

EJERCICIOS PROPUESTOS

1. Escribir un programa que almacene en una lista los números del 1 al 10 y los muestre por pantalla en orden inverso separados por comas
2. Escribir un programa que determine si una frase es un palíndromo, ejemplo, “yo hago yoga hoy”
3. Escribir un programa que pida al usuario una palabra y muestre por pantalla el número de veces que contiene cada vocal
4. Escribir un programa que pida al usuario una cadena y verifique que todas las vocales están presentes o no, y si no muestre que vocales no están
5. Escribir un programa que pida un número y devuelva una lista con sus divisores
6. Escribir un programa que cree una cesta de la compra. El programa debe preguntar el artículo y su precio y añadir el par al diccionario, hasta que el usuario decida terminar. Después se debe mostrar por pantalla la lista de la compra y el coste total
7. Escribir un programa que cree un diccionario de traducción español-inglés. El usuario introducirá las palabras en español e inglés separadas por dos puntos, y cada para <palabra>:<traducción> separados por comas. Después pedirá una frase en español y utilizará el diccionario para traducirla palabra a palabra. Si una palabra no está en el diccionario debe dejarla sin traducir
8. Escribe un programa que solicite por teclado los extremos de un intervalo y después evalúe el polinomio $P(x) = x^3 - 4x^2 + 8$ en 10 puntos equiespaciados en dicho intervalo. El programa debe imprimir dos columnas: x, P(x).