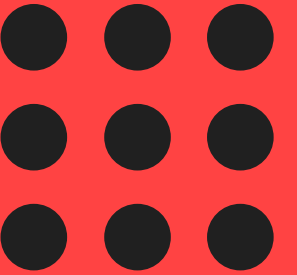


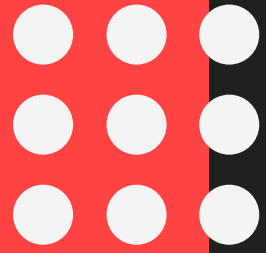
3001A-V22 ESTRUCTURA DE DATOS Y ALGORITMOS



# RECURSIVIDAD

Profesor: Rafael Pérez Aguirre





# ¿Qué es una iteración?

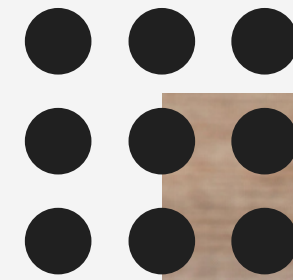
LA ITERACIÓN OCUPA MENOS MEMORIA QUE  
LA RECURSIÓN.



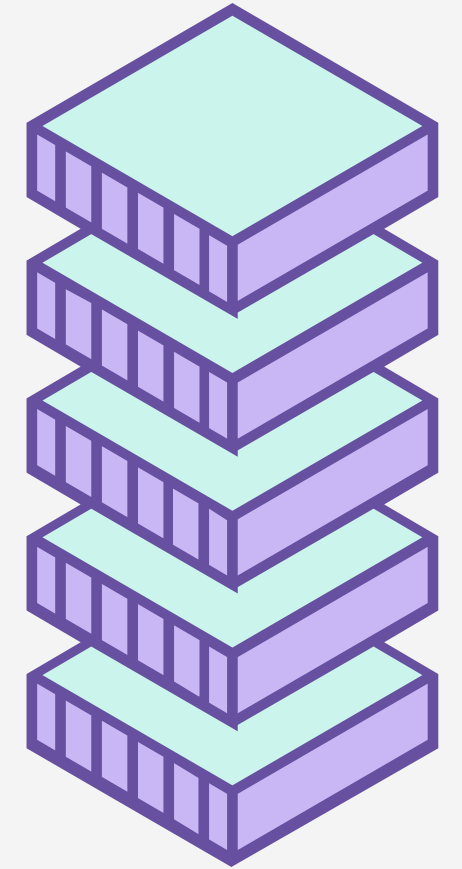
# Recursividad

**La recursividad es el proceso de definir una función en términos de sí misma**

Es una alternativa diferente para implementar estructuras de repetición (ciclos)



# ESTRUCTURA DE UNA FUNCIÓN RECURSIVA



- **Caso base**

Una solución simple para un caso particular (puede haber más de un caso base).

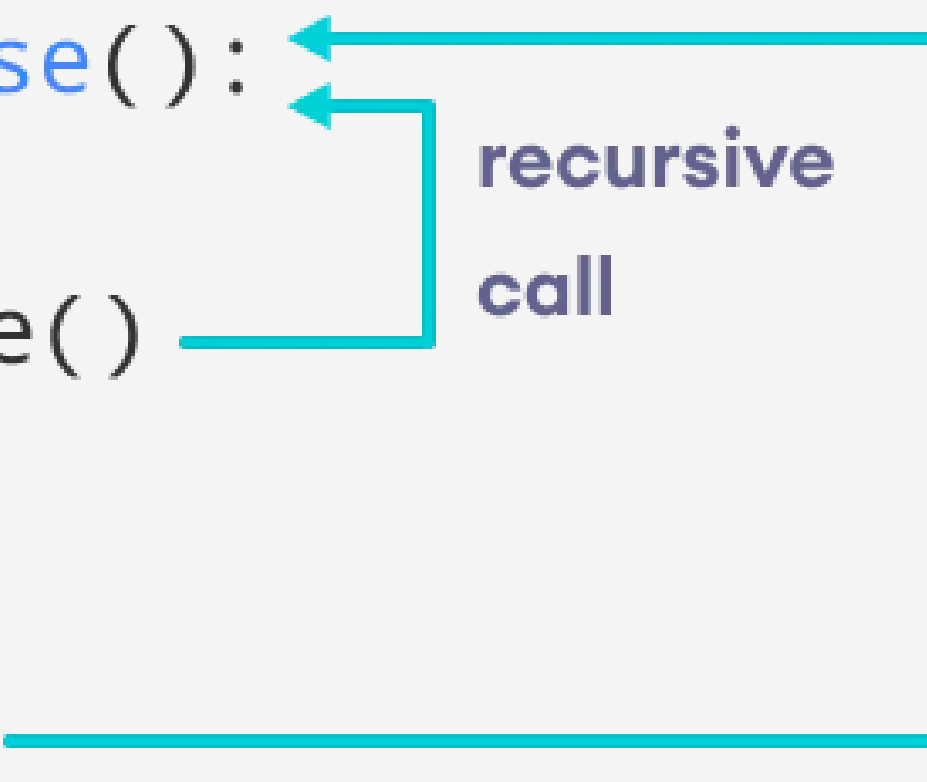
- **Caso general**

El caso general es para el cual la solución es expresada en términos de una pequeña versión de sí misma.

# Función recursiva en Python

Función que se llama a si misma

```
def recurse():  
    ...  
    recurse()  
    ...  
  
recurse()
```



recursive call

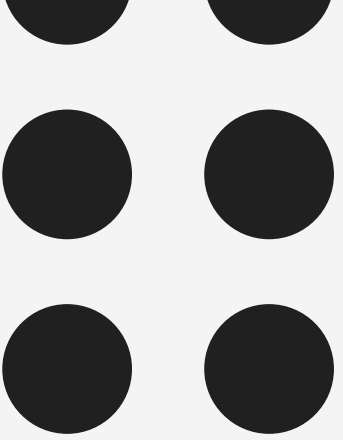
## Ventajas

- Las funciones recursivas hacen que el código se vea limpio y elegante.
- Una tarea compleja se puede dividir en subproblemas más simples usando recursividad.
- La generación de secuencias es más fácil con la recursividad que con alguna iteración anidada.

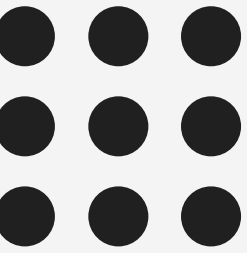


## Desventajas

- A veces, la lógica detrás de la recursividad es difícil de seguir.
- Las llamadas recursivas son costosas (ineficientes) ya que ocupan mucha memoria y tiempo.
- Las funciones recursivas son difíciles de depurar.



# Análisis



## Factorial

$$3! = 3 \cdot 2 \cdot 1 = \underbrace{\quad}_6 \downarrow \\ 6 = 6$$

$$5! = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 \\ \underbrace{\quad}_{20} \\ \underbrace{\quad}_{60} \\ \underbrace{\quad}_{120} \\ \boxed{120}$$

Recursividad

$$5! = 5 \cdot 4! \longrightarrow 5 \cdot 4! = 5 \cdot 4 \cdot 3! - \text{ } \\ 5 \cdot 4 \cdot 3!$$