Recursividad

IITA 2023

Primero…un repaso:

- Python…¿Qué es?
- SOFTWARE
- HARDWARE
- LENGUAJE DE PROGRAMACIÓN
- PROGRAMA
- TIPO DE DATO
- VARIABLE

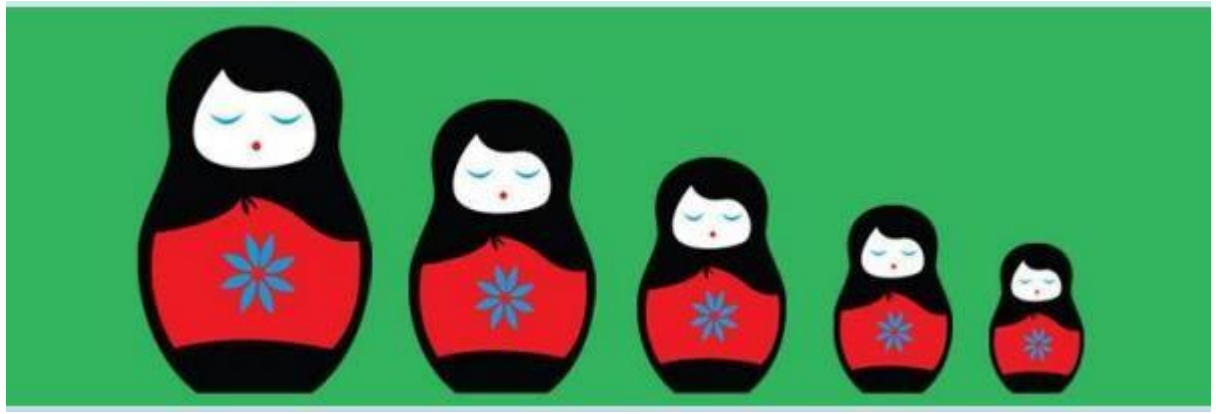
- Estructuras:
  - Selectivas
    - Simples
    - Múltiples
  - Repetitivas

- Listas
  - Operaciones
- Tuplas
- Diccionarios
- Conjuntos

- Funciones:
  - Ámbito de las variables

Recursividad

- Permite la resolución de problemas evolutivos al dividir un problema P en problemas p más pequeños, pero de la misma naturaleza que P
- Un objeto es recursivo si figura dentro de su definición. Se presenta como una alternativa a la iteración o repetición de funciones y/o sentencias,



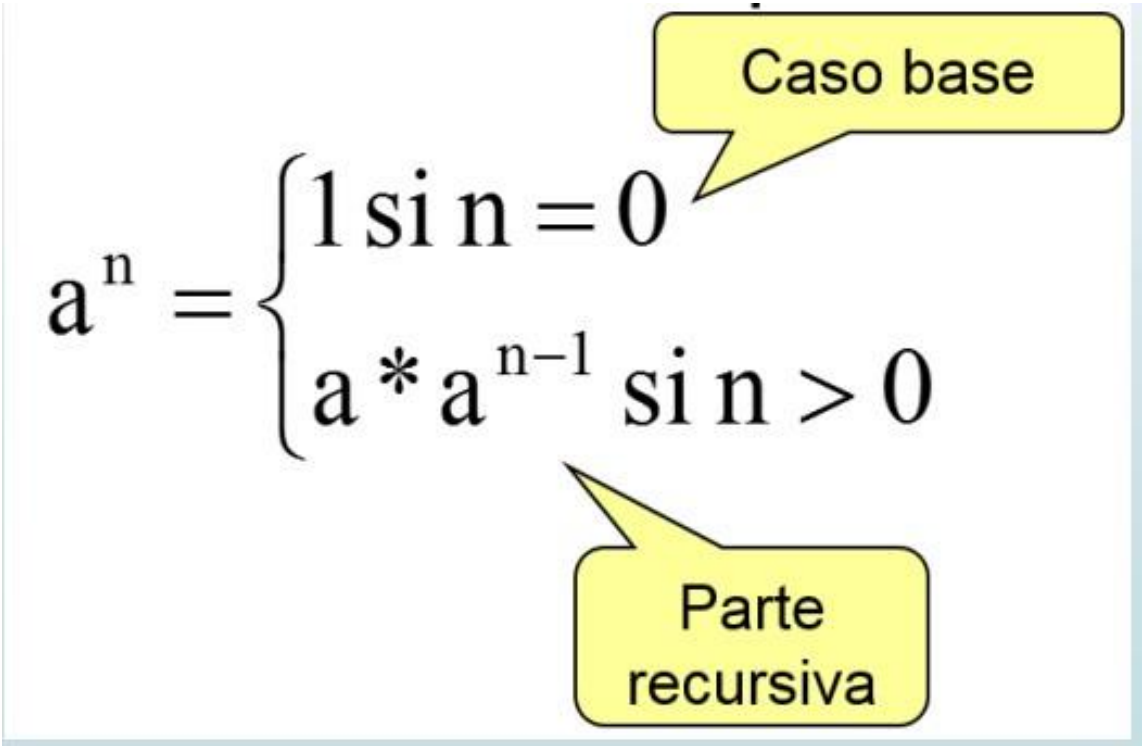
Recursividad

- Se caracterizan por su “simplicidad y elegancia”
  - Esto es un punto a favor contra la dificultad y poca legibilidad de una función repetitiva
- Conducen a soluciones más fáciles de leer y comprender
  - Conllevan un costo de tiempo de ejecución y de memoria en la computadora

```
3  def digitos(num, dig):
4      if num == 0:
5          return dig
6      else:
7          return digitos(num // 10, 1 + dig)
8
9  #iteracion
10
11 def digitos1(num):
12     dig = 0
13     num == abs(num)
14     while num !=0:
15         num //= 10
16         dig += 1
17     return dig
18
19
20 print(digitos(1337,0))
```

Recursividad

- Se establece un "**caso base**" o "caso de salida" para establecer el final del proceso recursivo, de esta manera puede ser una solución más simple y rápida para algunos tipos de problemas
- Por ejemplo, para calcular la potencia N de un número dado



Recursividad

$$a^n = \begin{cases} 1 \text{ si } n = 0 \\ a * a^{n-1} \text{ si } n > 0 \end{cases}$$

Caso base

Parte recursiva

```
def potencia(base, exp):
    ans=1
    if exp==0:
        return ans
    else:
        while exp>=1:
            ans*=base
            exp-=1
        return ans
```

Recursividad

- Caso Base:
  - Son aquellos que para su solución no requieren utilizar la función que se está definiendo
- Caso Recursivo:
  - Son aquellos que sí que requieren utilizar la función que se está definiendo

→ Por ejemplo: factorial de un número

El factorial de N es:
1x2x3x4x5 … x N

```
def factorial(numero):
    if(numero == 1):
        return 1
    else:
        return numero * factorial(numero - 1)

print(factorial(12))
```