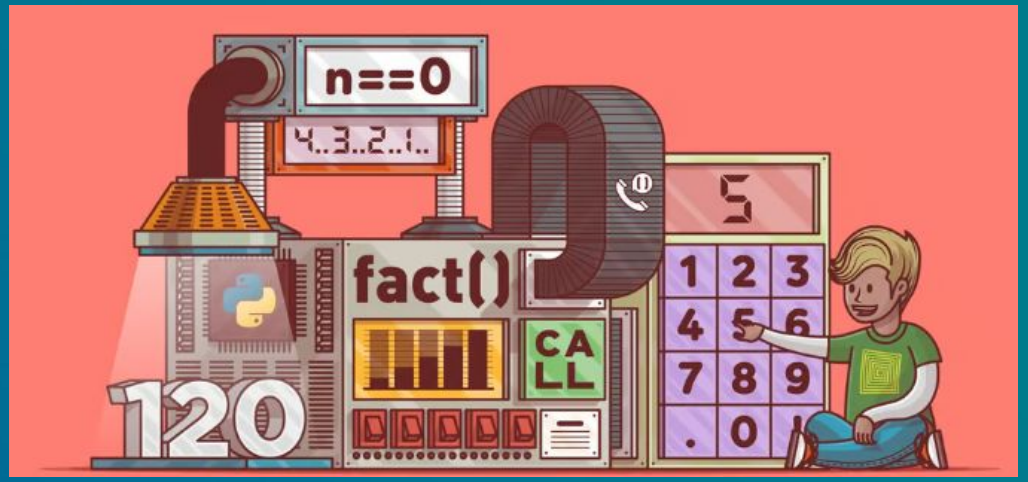
Funciones recursivas



Las muñecas rusas (matrioshkas)

Las matrioshkas son un conjunto de muñecas que se colocan una dentro de la otra. Cada muñeca se abre para revelar otra más pequeña en su interior, y así sucesivamente, hasta llegar a la más pequeña que no se abre.

A partir de esta analogía, podemos analizar el concepto de **recursividad**.

Recursividad

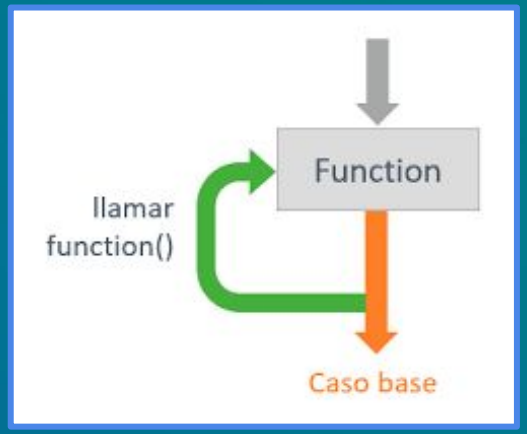
Cuando un proceso se define en términos de sí mismo hablamos de recursividad. Básicamente un problema podrá ser resuelto de forma recursiva si la solución se puede expresar en términos de si misma. Para poder obtener esta solución, deberá resolverse el mismo problema sobre un conjunto de datos de entrada menor.

¿En qué trabajás? Estoy intentando arreglar los problemas que creé cuando intentaba arreglar los problemas que creé cuando intentaba arreglar los problemas que creé.

Y así nació la recursividad.

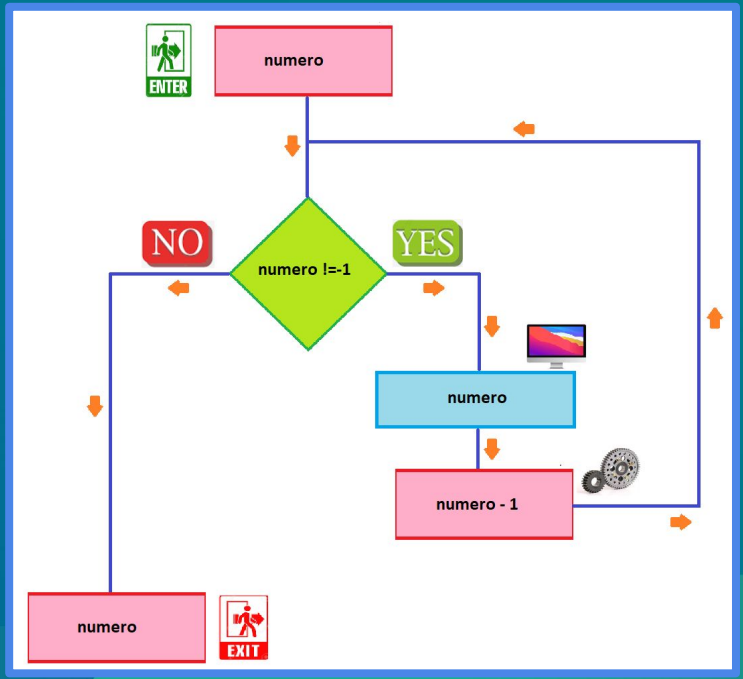
Funciones recursivas

La recursividad aplicada al mundo de la programación nos permite resolver tareas en donde las mismas pueden ser divididas en **sub tareas cuya funcionalidad es la misma**. Una función recursiva es aquella que está definida en función de si misma, por lo que se llama repetidamente a sí misma **hasta llegar a un punto de salida**.



Pongámonos a prueba

El siguiente diagrama de flujo muestra una cuenta regresiva desde un número ingresado hasta el 0. Planteemos una solución desde un punto de vista convencional (con una estructura repetitiva) y desde un punto de vista recursivo



Factorial de un número

El factorial de un número entero positivo se define como el producto de todos los números enteros positivos desde la unidad hasta el número definido.

Matemáticamente definimos al factorial:

$$n! = n * (n - 1)!$$
$$\text{Si } n >= 0$$

Funciones recursivas

5! = 5*4*3*2*1 = 5 * 4! = 120

4! = 4*3*2*1 = 4 * 3! = 24

3! = 3*2*1 = 3 * 2! = 6

2! = 2*1 = 2 * 1! = 2

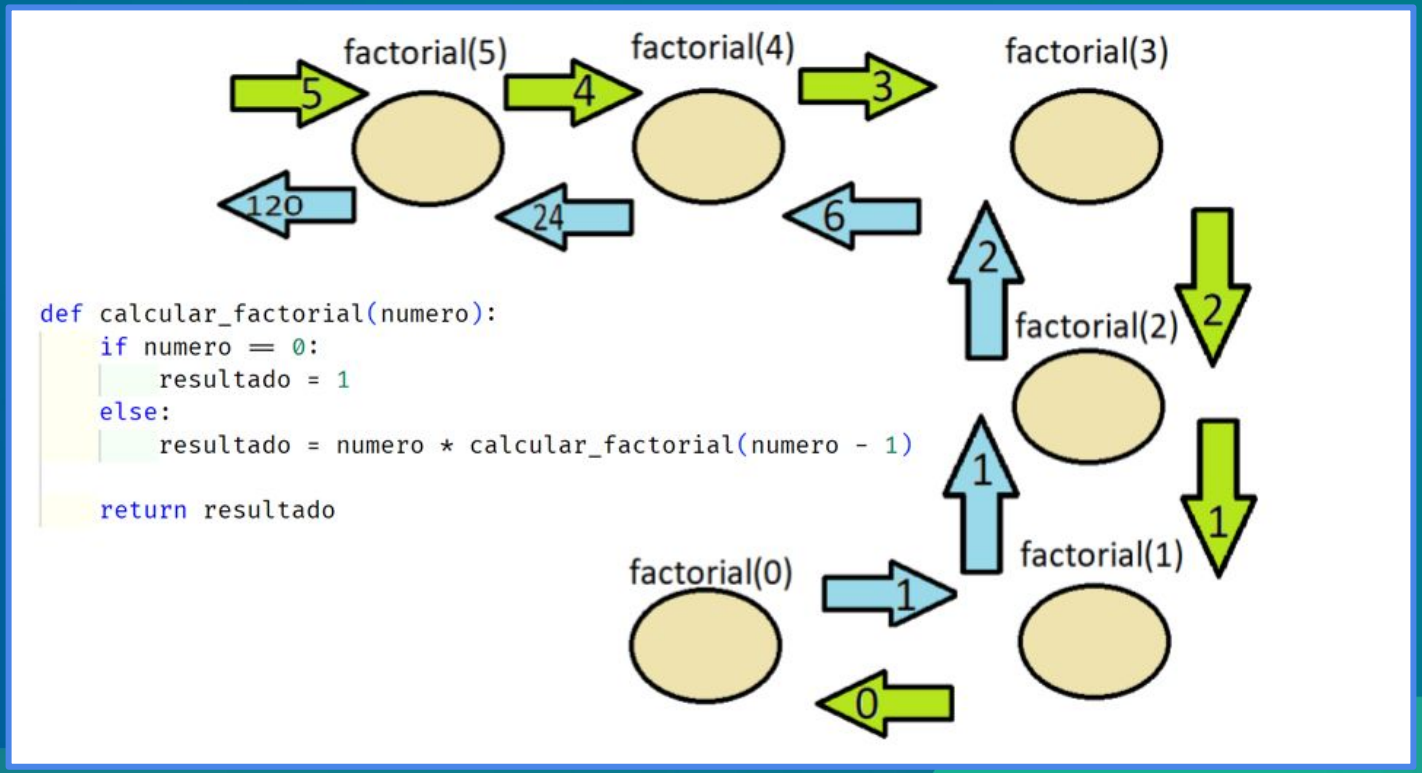
1! = 1

0! = 1

Programación I

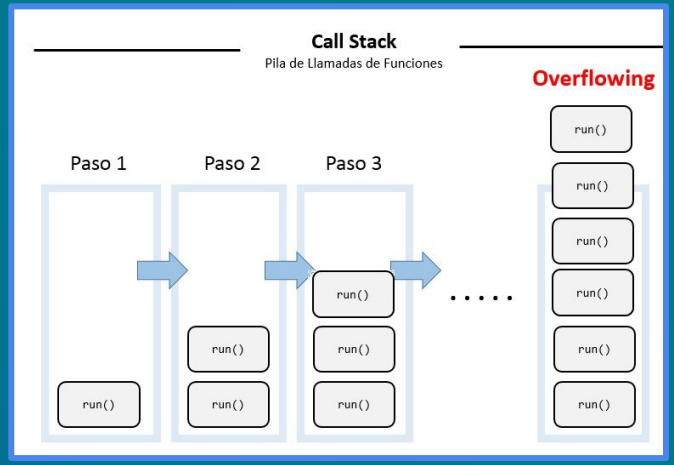
Scarafilo Germán - Editores: Fernández M - Pavlov Valeriy

Funciones recursivas



Pero ... ¿qué pasa en memoria?

- La gestión de memoria en las funciones recursivas se realiza mediante la pila de llamadas.
- Cada vez que se llama a una función recursiva, se agrega una nueva instancia de esa función a la pila de llamadas.
- A medida que la función se llama recursivamente se acumulan mas y mas instancias en la pila.
- Cuando se alcanza la condición de salida, las funciones comienzan a desapilarse, en caso de que dicha condición no se cumpla y se llegue a cierto límite (llamado el límite de recursión), python lanzará un error de desbordamiento de pila (**RecursionError**).



Ventajas

- **Simplicidad conceptual**: algunos problemas se pueden expresar de manera más clara y concisa.
- **Solución elegante para problemas recursivos**: hay problemas que naturalmente se prestan a una solución recursiva, como árboles o la división y conquista.
- **Facilidad de mantenimiento**: en algunos casos, el código recursivo puede ser más fácil de entender y mantener que su equivalente repetitivo.

Desventajas

- **Consumo de memoria**: cada llamada recursiva agrega una nueva instancia de la función a la pila de llamadas, esto puede consumir una cantidad significativa de memoria, especialmente para problemas con profundidad recursiva.
- **Eficiencia**: en general las funciones recursivas pueden ser menos eficientes que sus equivalentes iterativos debido al costo adicional de gestionar la pila de llamadas y la posibilidad de realizar cálculos redundantes.
- **Límite de recursión**: python tiene un límite en la profundidad de recursión, por lo que las funciones recursivas pueden no ser adecuadas para problemas con grandes profundidades recursivas.