Recursividad

Se denomina llamada recursiva (o recursividad), a aquellas funciones que en su algoritmo, hacen referencia sí misma.

Las llamadas recursivas suelen ser muy útiles en casos muy puntuales, pero debido a su gran factibilidad de caer en iteraciones infinitas, deben extremarse las medidas preventivas adecuadas y, solo utilizarse cuando sea estrictamente necesario y no exista una forma alternativa viable, que resuelva el problema evitando la recursividad.

Python admite las llamadas recursivas, permitiendo a una función, llamarse a sí misma, de igual forma que lo hace cuando llama a otra función.

```
def jugar(intento=1):
    respuesta = input("¿De qué color es una naranja? ")
    if respuesta != "naranja":
        if intento < 3:
            print ("\nFallaste! Inténtalo de nuevo")
            intento += 1
            jugar(intento)
        else:
            print("\nPerdiste!")
    else:
        print("\nGanaste!")
jugar()
```

```
def calc_factorial(x):
    if x == 1:
        return 1
    else:
        return (x * calc_factorial(x-1))

num = 4
print("The factorial of", num, "is", calc_factorial(num))
```

En el ejemplo de arriba, calc_factorial() es una función recursiva dado que se llama a si misma. Cuando llamamos esta función usando un entero positivo, se llamará a si misma hasta que el numero disminuya.
Cada función llama a los múltiplos con el factorial 1 hasta que el número es igual a uno, esta llamada recursiva se puede explicar en los siguientes pasos.

```
calc_factorial(4)              # 1er llamada con 4
4 * calc_factorial(3)          # 2da llamada con 3
4 * 3 * calc_factorial(2)      # 3er llamda con  2
4 * 3 * 2 * calc_factorial(1)  # 4ta llamada con 1
4 * 3 * 2 * 1                  # return de la cuarta llamada con number=1
4 * 3 * 2                      # return de la tercer llamada
4 * 6                          # return de la segunda llamada
24                             # return de la primer llamada
```

Nuestra recursividad termina cuando el numero queda en 1. Eso es llamado la condición base.

Cada función recursiva debe tener una condición base que detenga la recursividad o si no la función se llamará una infinidad de veces.

Ventajas de la recursividad
Las funciones recursivas hacen ver el código limpio y elegante.
Una tarea compleja puede romperse en pequeños problemas usando recursión.
La generación de una secuencia es mucho más fácil usando recursión que usando loops anidados.

Desventajas de la recursión
A veces es difícil seguir la lógica detrás de la recursión
Las llamadas recursivas son ineficientes ya que toman mucha memoria y tiempo.
Es difícil encontrar debuggear una función recursiva.