

Programación I

Recursividad

Recursividad

La recursividad es un concepto en programación en el que una función se llama a sí misma directa o indirectamente para resolver un problema.

Es especialmente útil para resolver problemas que pueden ser descompuestos en subproblemas más pequeños y similares al problema original.

La recursividad se basa en dos partes principales: **el caso base y el caso recursivo.**

El caso base y el caso recursivo

Caso Base:

- Es la condición que detiene la recursión.
- Define el escenario más simple o básico que no requiere una llamada recursiva.
- Es crucial para evitar que la función se llame indefinidamente.

Caso Recursivo:

- Es donde se realiza la llamada recursiva a la función dentro de sí misma.
- Se encarga de dividir el problema original en subproblemas más pequeños.
- Se combina con el resultado de la llamada recursiva para resolver el problema original.

Calculo factorial usando recursividad

```
ej06ClaseEjemplo.java U
src > CursoJava

public class RecursividadFactorial {

    // Método para calcular el factorial de un número de forma recursiva
    public static int factorial(int n) {
        // Caso base: si n es 0 o 1, el factorial es 1
        if (n == 0 || n == 1) {
            return 1;
        } else {
            // Caso recursivo: n * factorial(n - 1)
            return n * factorial(n - 1);
        }
    }

    public static void main(String[] args) {
        int numero = 5;
        int resultado = factorial(numero);
        System.out.println("El factorial de " + numero + " es: " + resultado);
    }
}
```

En este ejemplo, el método factorial calcula el factorial de un número n . Si n es 0 o 1, el método devuelve 1 (caso base). De lo contrario, realiza una llamada recursiva multiplicando n por el factorial de $n - 1$ (caso recursivo). Este proceso se repite hasta que se alcanza el caso base.