# Explicação Didática sobre Funções em Java

## Funções em Java (também chamadas de métodos)

Imagine uma função como uma caixinha de código que você monta para fazer uma tarefa específica — como somar dois números ou mostrar uma mensagem na tela. Essa caixinha pode ser usada várias vezes no seu programa, sempre que você quiser repetir essa tarefa.  
  
Estrutura de uma função:  
tipoDeRetorno nomeDoMetodo(parâmetros) {  
 // código que executa a tarefa  
 return valor; // se não for void  
}  
  
- tipoDeRetorno: é o tipo do valor que a função vai devolver (ex: int, String, void).  
- nomeDoMetodo: o nome que você dá para essa caixinha.  
- parâmetros: os valores que você envia para a função realizar a tarefa.  
- return: devolve um resultado se for o caso (funções void não usam return com valor).

## Função vs. Procedimento

Função: devolve um resultado.  
Exemplo:  
int soma(int a, int b) {  
 return a + b;  
}  
  
Procedimento: executa uma ação, mas não retorna nada. Usamos void.  
Exemplo:  
void imprimirMensagem(String mensagem) {  
 System.out.println(mensagem);  
}

## Passagem de Parâmetros

É como você entrega informações para as funções. Isso pode ser feito de dois jeitos:  
  
1. Passagem por Valor (tipos primitivos):  
Você entrega uma cópia do valor. Se a função mudar o valor, não afeta o valor original fora dela.  
int numero = 5;  
modificar(numero); // muda só dentro do método  
  
2. Passagem por Referência (objetos):  
Você entrega um endereço do objeto, então a função pode modificar o original.  
Pessoa p = new Pessoa();  
p.nome = "João";  
modificarPessoa(p); // vai mudar o nome para "Maria"  
  
Resumo:  
- Primitivos (int, char, etc.): cópia → não muda o original.  
- Objetos: referência → pode mudar o original.

## Exemplo Completo de Passagem de Parâmetros

Você tem dois métodos:  
  
public void modificarValorPrimitivo(int numero) {  
 numero = numero + 10;  
 System.out.println("Dentro do método: " + numero);  
}  
  
public void modificarValorObjeto(Pessoa pessoa) {  
 pessoa.nome = "Maria";  
 System.out.println("Dentro do método: " + pessoa.nome);  
}  
  
Ao rodar o programa, você verá:  
- O número não muda fora da função.  
- O nome da pessoa muda para "Maria".

## Recursão

Recursão é quando uma função chama ela mesma.  
  
Pense como um espelho em frente ao outro: ele repete a imagem infinitamente — até um limite.  
  
Exemplo clássico: Fatorial  
int fatorial(int n) {  
 if (n == 0 || n == 1) return 1;  
 return n \* fatorial(n - 1);  
}  
  
Recursão resolve problemas quebrando-os em pedaços menores. Mas precisa ter uma condição de parada senão entra em loop infinito.

## Funções Lambda (Java 8 em diante)

Lambdas são funções mais curtas e modernas — úteis quando você quer passar um comportamento como se fosse uma variável.  
  
Sintaxe:  
(parâmetros) -> { corpo }  
  
Exemplo:  
x -> x \* 2  
  
São muito usadas com coleções para filtrar, transformar ou ordenar dados com pouco código.

## Por que tudo isso é útil?

- Modulariza o código: partes separadas e organizadas.  
- Evita repetição: escreve uma vez, usa sempre que precisar.  
- Facilita manutenção: mudar um método altera o comportamento em vários lugares.