



Tópico 1: Intro a creación y uso de métodos

Semestre: 01/16

No. Serie

Contenidos

- ¿Qué es un método?
- ¿Para qué sirven los métodos?
- ¿Cómo se crean y usan los métodos?

```
1 public class Calculos{
2
3     public int Suma(int a, int b){
4         return a + b;
5     }
6
7     public double Suma(double a, double b){
8         return a + b;
9     }
10
11     public long Suma(long a, long b){
12         return a + b;
13     }
14
15 }
```

¿Qué es un método?

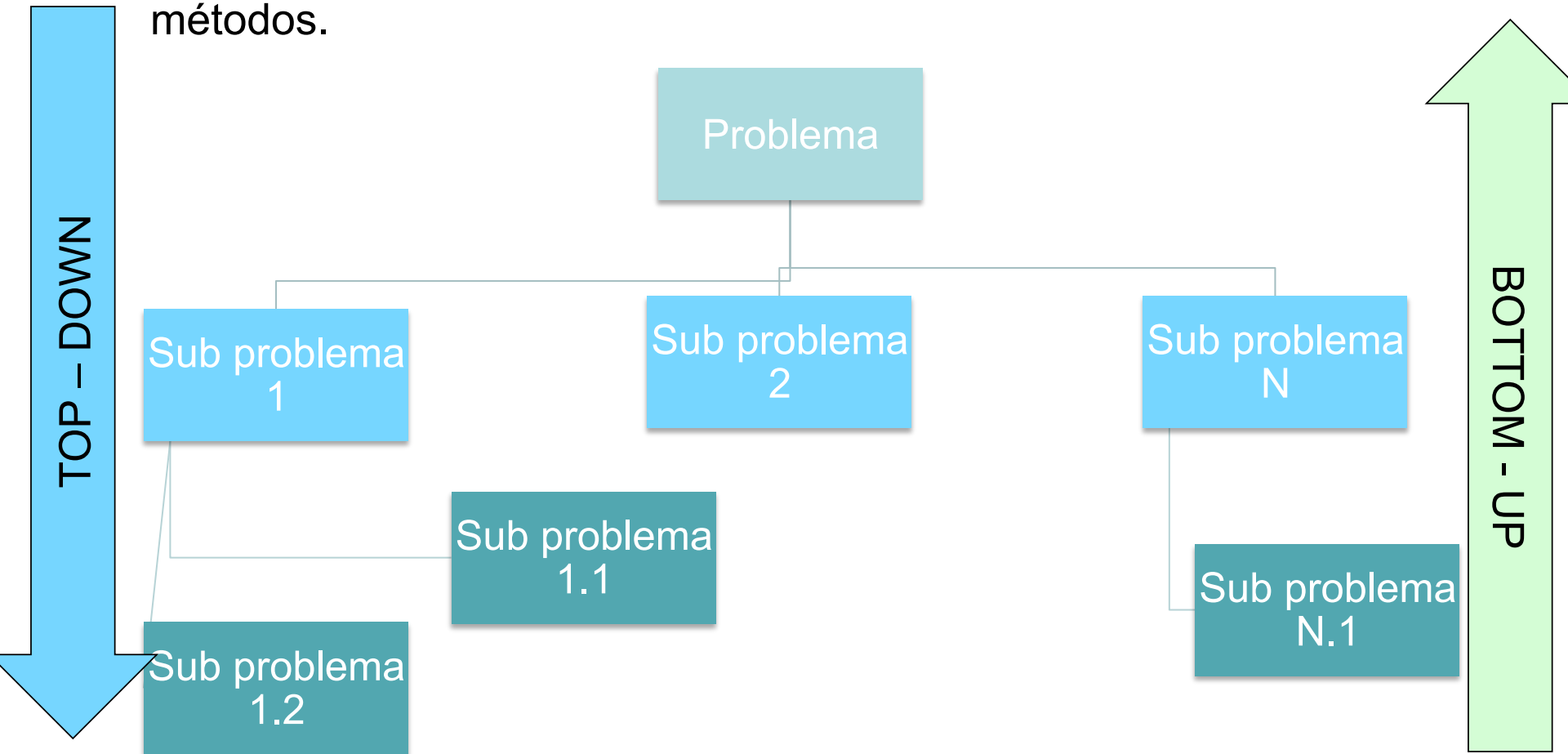
- Método es un **conjunto de líneas de código fuente** que realizan una **tarea específica y puede o no retornar un valor**.
- Tradicionalmente también se les ha conocido como:
 - Funciones
 - Procedimientos
- Para mantener los estándares seguidos por la POO, de ahora en adelante los llamaremos simplemente Métodos
- **OBS:**
 - LDP: Lenguajes de Programación
 - POO: Programación Orientada a Objetos

¿Para qué sirven los métodos?

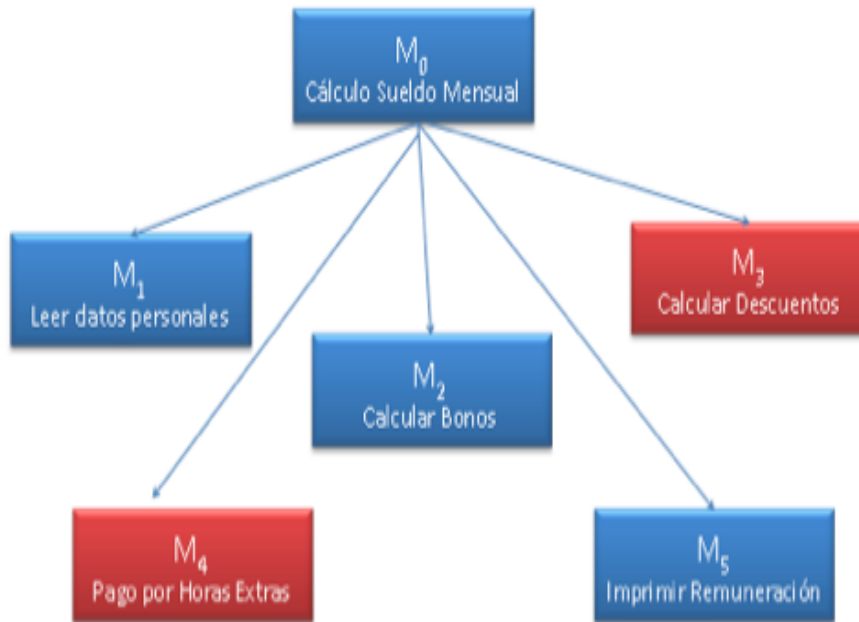
- Son utilizadas para **descomponer** grandes problemas en **tareas simples** y para implementar operaciones que son comúnmente utilizadas durante un programa.
- Dividir el problema en problemas más pequeños (**Dividir para vencer**)
 - Problemas pequeños resultan más fáciles de entender y mantener (**Pensando en el futuro**)
- Tengo varias veces un mismo problema cuya solución puedo repetir (**Pensando en reutilizar**)
 - **Reducen** la cantidad de código (soluciones “más limpias”)

Diseño de soluciones basadas en métodos

- Metodologías **Top-Down** y **Bottom-Up** se sustentan en el uso de métodos.

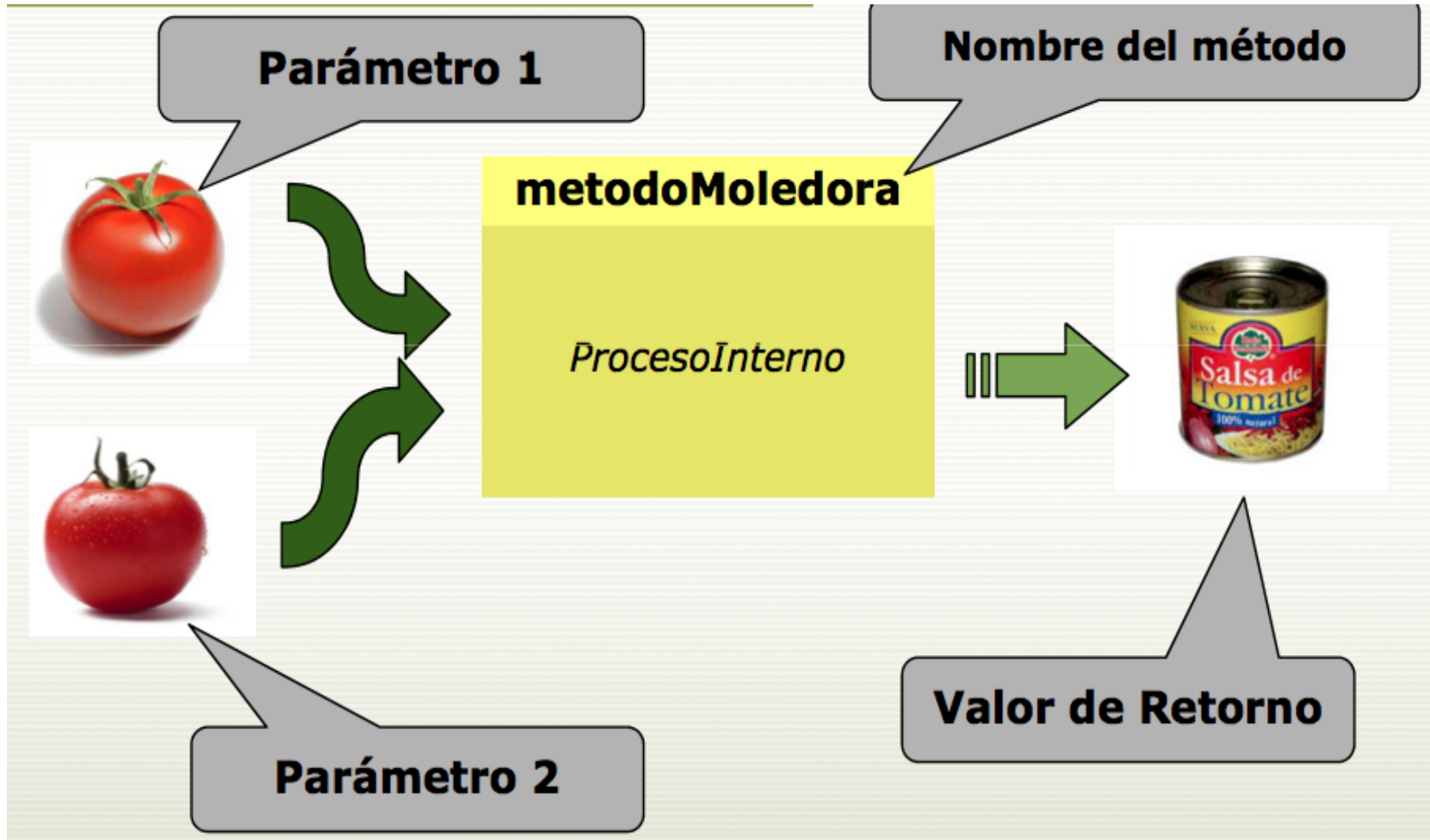


Diseño de soluciones basadas en métodos

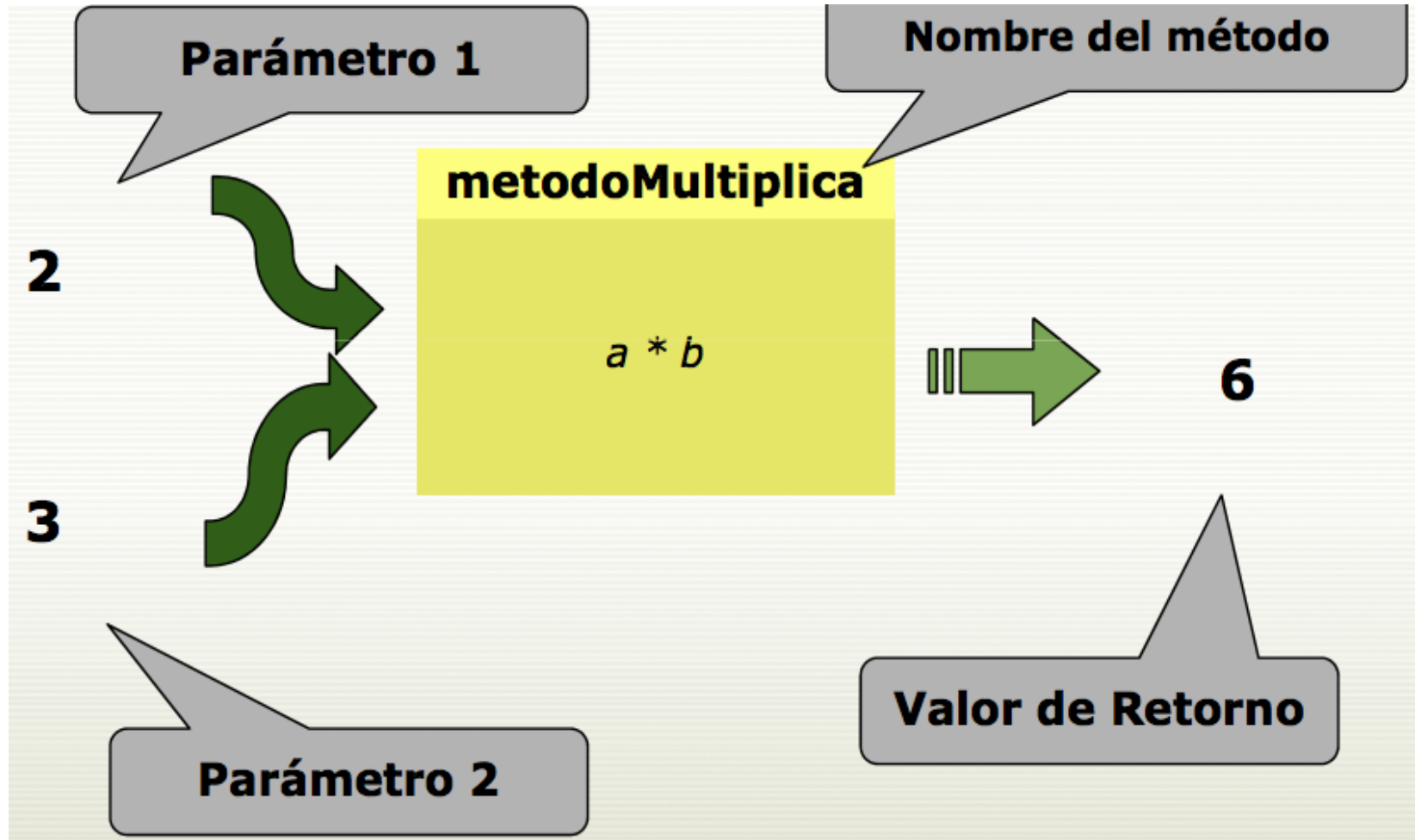


- Un problema se puede descomponer en problemas más pequeños.
- Algunos de estos pueden ya estar desarrollados (reutilización).
- Incluso pudiendo haber sido desarrollado por terceros, p. ej. M3 y M4 (externalización del desarrollo).

Ilustrando el concepto de Método



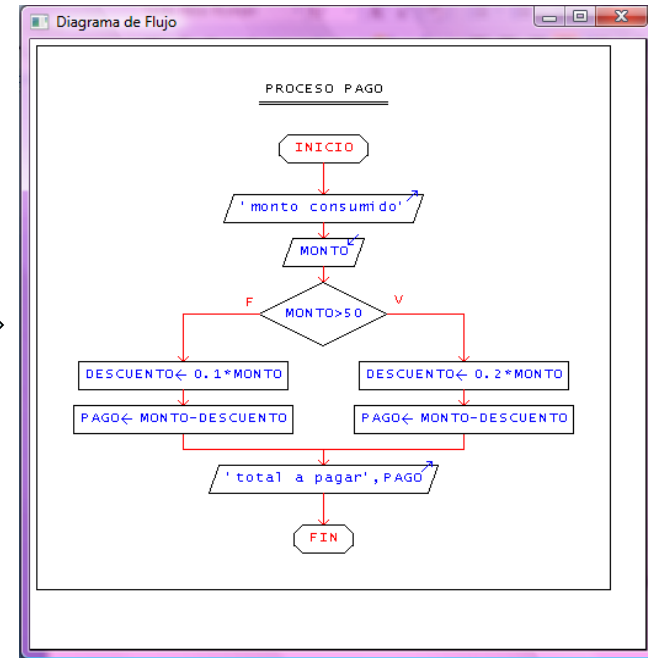
Ilustrando el concepto de Método



Métodos y control del flujo de un programa

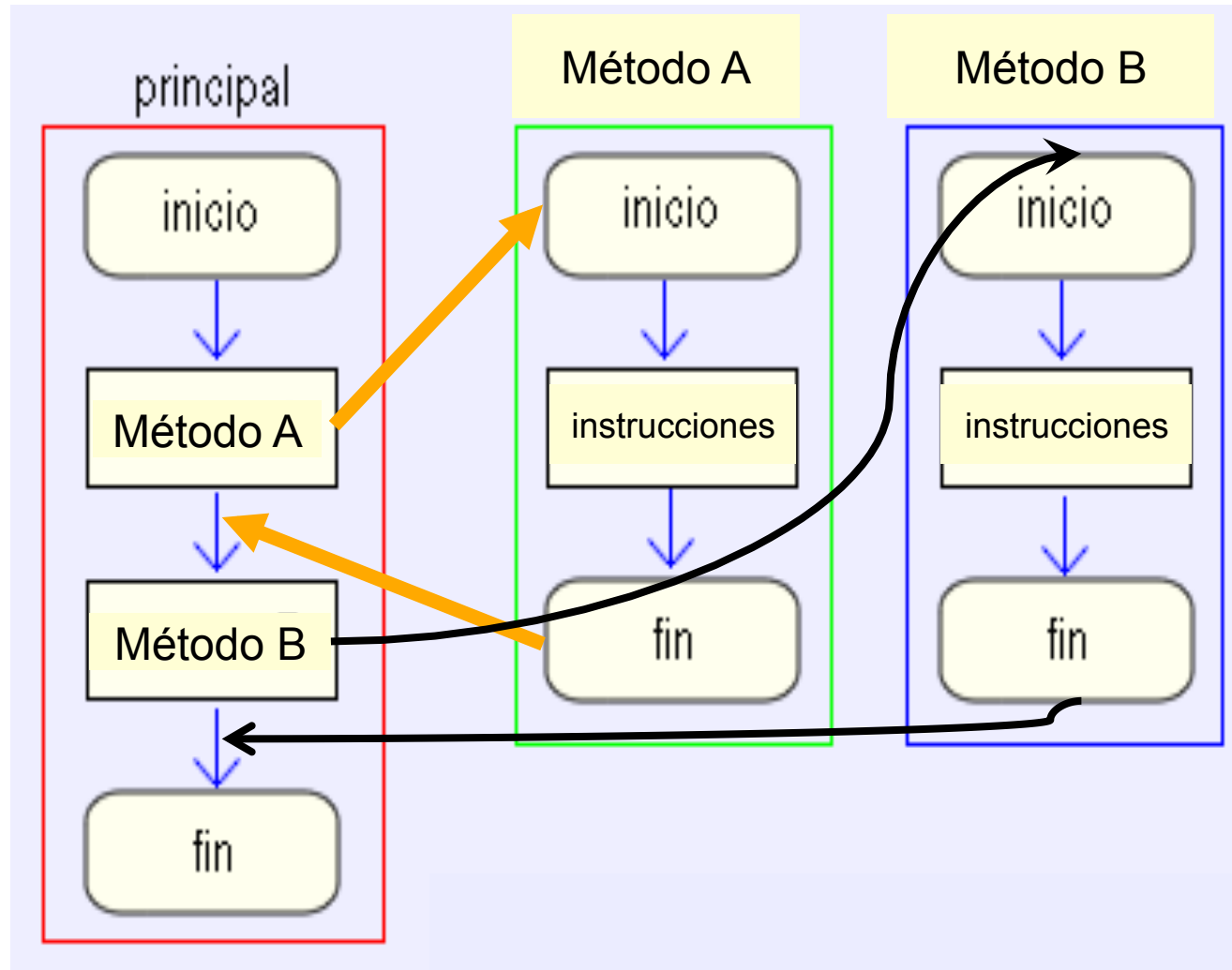
- Si un método es invocado desde alguna sección de un programa, se le pasa el control del programa al método.
- Una vez que éste finalizó con su tarea (y posiblemente haya devuelto un resultado) el control del programa es devuelto al punto desde el cual el método fue invocado.

Comparar con el flujo de un programado sin uso de métodos



Métodos y control del flujo de un programa

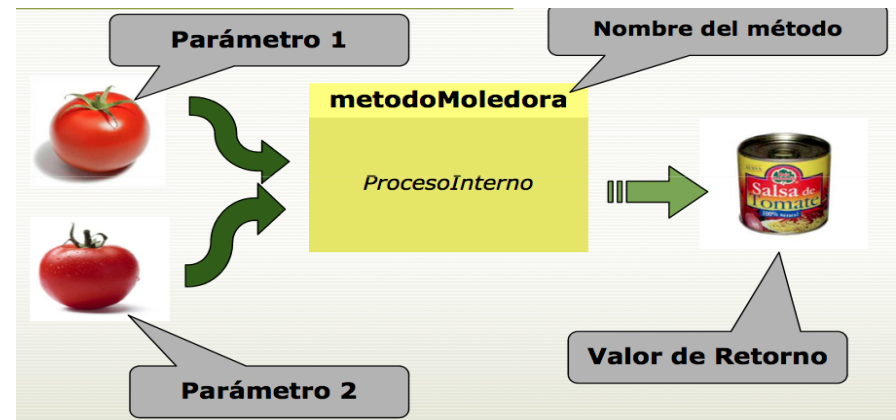
- Lo anterior pero de forma gráfica, considerando:
 - Un programa principal
 - 2 métodos A y B
 - Primero se invoca al método A
 - Luego se invoca al método B y finaliza el programa



¿Cómo se crean y usan los métodos?

- Ejemplo 1
 - Considere el caso en el cual se desea crear un método que implemente el código fuente que retorne el valor de un número real al cuadrado que recibe como parámetro.

```
real cuadrado (real parametro) {  
    retorne parametro*parametro;  
}
```



Definiendo un método

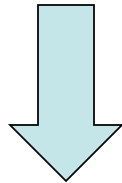
- A partir de lo anterior, podemos notar que debemos saber:
 - ¿Qué **debe hacer** nuestro Método?
 - ¿Qué **nombre** tendrá nuestro Método?
 - ¿Qué **parámetros (nombre y tipo)** usará nuestro Método?
 - ¿Qué **tipo** de valor devolverá nuestro Método?

```
tipo_retorno nombre_Método ( [Parámetros] ) {  
    Declaración variables locales; //si se necesitan  
    Instrucciones (Cuerpo del método);  
    Valor de retorno (del tipo_retorno); //si existe  
}
```

Definiendo un método – ejemplo con Java

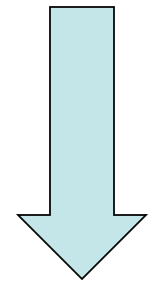
- Método que implemente el código fuente que retorne el valor de un número real al cuadrado que recibe como parámetro.

```
real numCuadrado (real parametro) {  
    retorne parametro*parametro;  
}
```



```
double numCuadrado (double parametro) {  
    return parametro*parametro;  
}
```

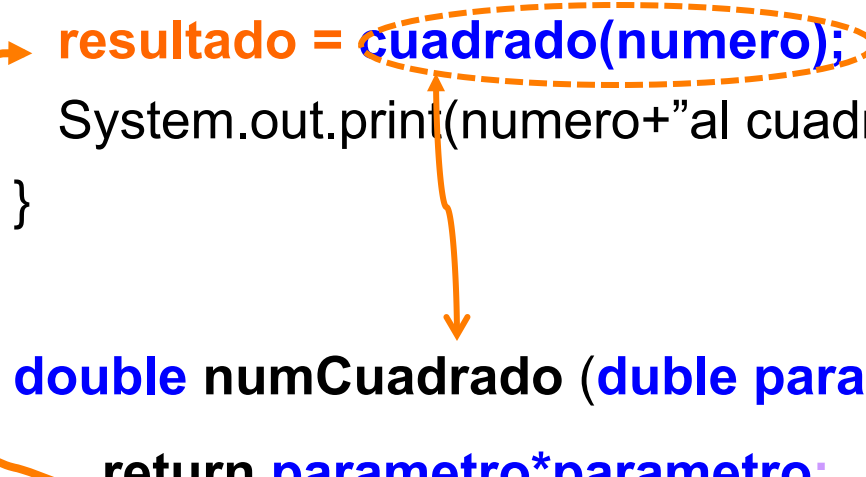
Diseño del método



**Implementación del
Método con Java**

Usando un método

```
import ...  
public class ejemplo(...) {  
    double numero, resultado;  
    leer numero // leer y convertir tipo como ud. estime  
    resultado = cuadrado(numero);  
    System.out.print(numero+"al cuadrado es:" + resultado);  
}  
  
double numCuadrado (duble parametro) {  
    return parametro*parametro;  
}
```



Métodos - Tipos y retornos

```
//Sin parámetros y sin retorno
public void miMetodo( ){

}

//Sin parámetros, retorna un arreglo de enteros
public int[] miMetodo( ){

}

//Parámetros un arreglo de entero y un arreglo float bidim.
public String[] miMetodo(int[] a, float[][] b){

}

//Parámetros arreglo String y un String, retorna un entero
public int miMetod( String[] a, String b ) {

}
```

Métodos y ámbito de las variables

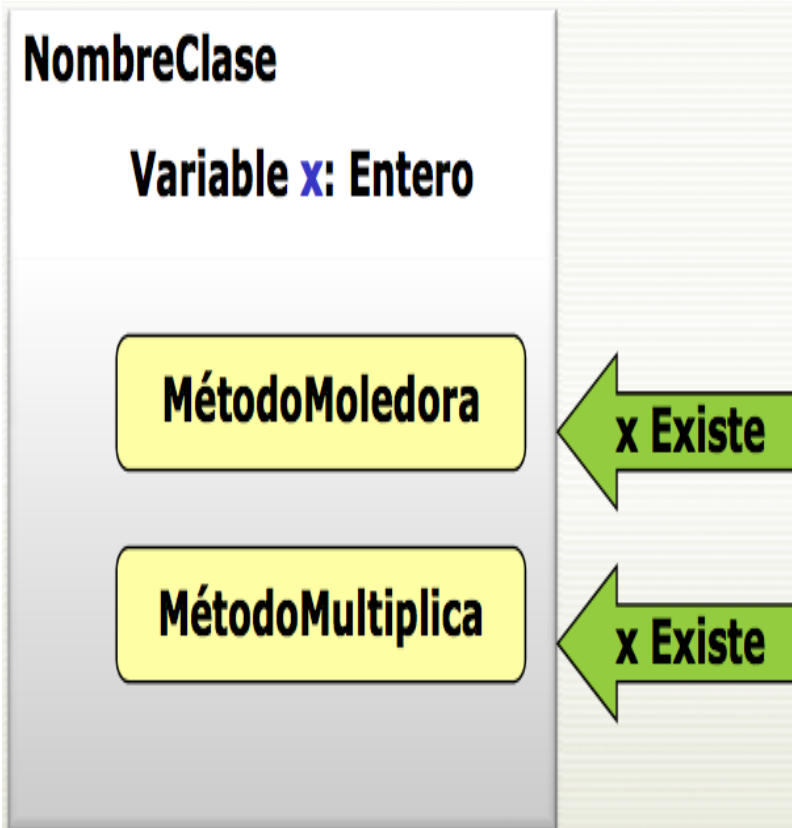
- Dependiendo de donde se defina y use una variable se habla del ámbito de las variables.
 - Variables globales
 - Variables locales

Variables globales: Se definen en el contexto global y se las llama variables de clases o atributos, mismas que pueden utilizarse en cualquier método de la clase.

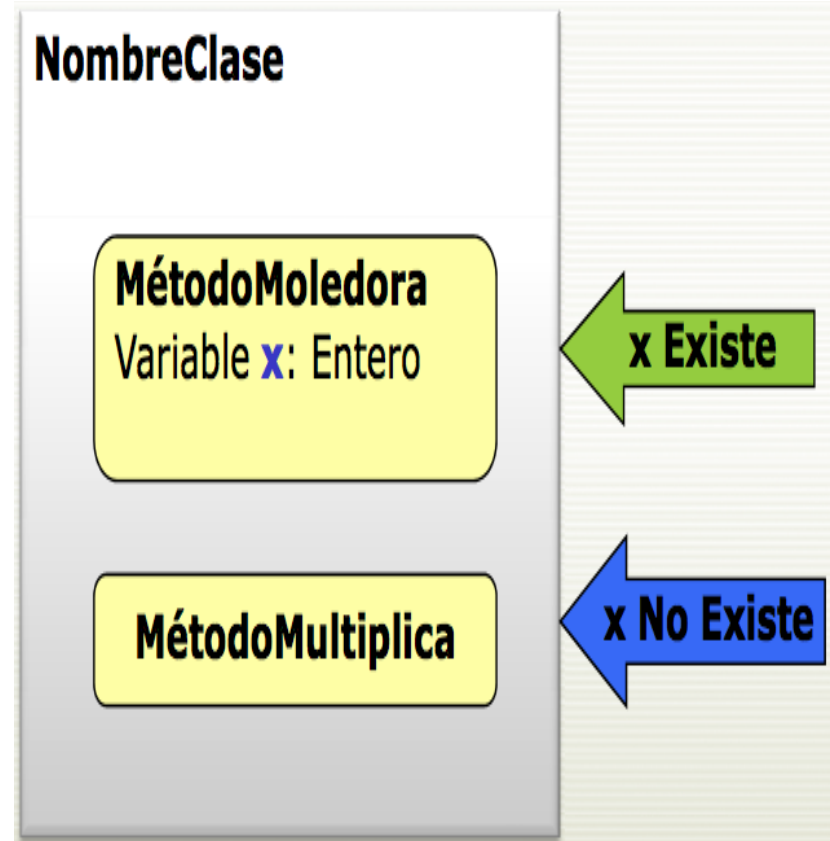
Variables Locales: Sólo se definen en cada método, por lo tanto sólo pueden usarse dentro del método que las declara.

Métodos y ámbito de las variables

Variable global



Variable local



Consideraciones para uso de Métodos

- Antes de empezar a “teclear”, discutamos:
 - ¿Cuántos y cuáles métodos debe tener el programa?
 - ¿Qué tipo debe tener c/u de los métodos?
 - ¿Qué parámetros requiere c/u de los métodos?
 - ¿Qué resultado debe retornar c/u de los métodos?



Ejercicios

1. Construya el ejemplo de “Hola Mundo con métodos”
 1. Su solución debe contener un método llamado saludo
 2. Dicho método debe mostrar el mensaje “Hola Mundo” por pantalla.

OJO! Antes de empezar a “teclear”, discutamos

¿Qué tipo debe tener el método?

¿Qué parámetros requiere el método?

¿Qué debe retornar el método?

2. Construya una v0.2, donde el mensaje mostrado lo decide el usuario.

Ejercicios

3. Elabore un programa que permita ordenar un arreglo de 10 celdas.

El programa debe solicitar los datos del arreglo, ordenar los datos del arreglo y posteriormente mostrar por pantalla los números ordenados.

Defina e implemente los métodos necesarios, de acuerdo al contexto del problema.

Ejercicios

4. Construya una simulación de la calculadora usando métodos.

El usuario ingresa 2 números enteros

Selecciona una opción en un MENÚ y el programa le debe permitir realizar una, algunas o todas las operaciones definidas

Las operaciones definidas son: sumar, restar, multiplicar y dividir ambos números.

Las opciones además deben incluir la opción de Salir del programa y de ingresar 2 nuevos números.

Resumiendo

- Concepto de método y sus características
- Importancia del uso de métodos en la programación
- Partes que componen un método
- Diseño de una solución basada en métodos
- Implementación de métodos con Java