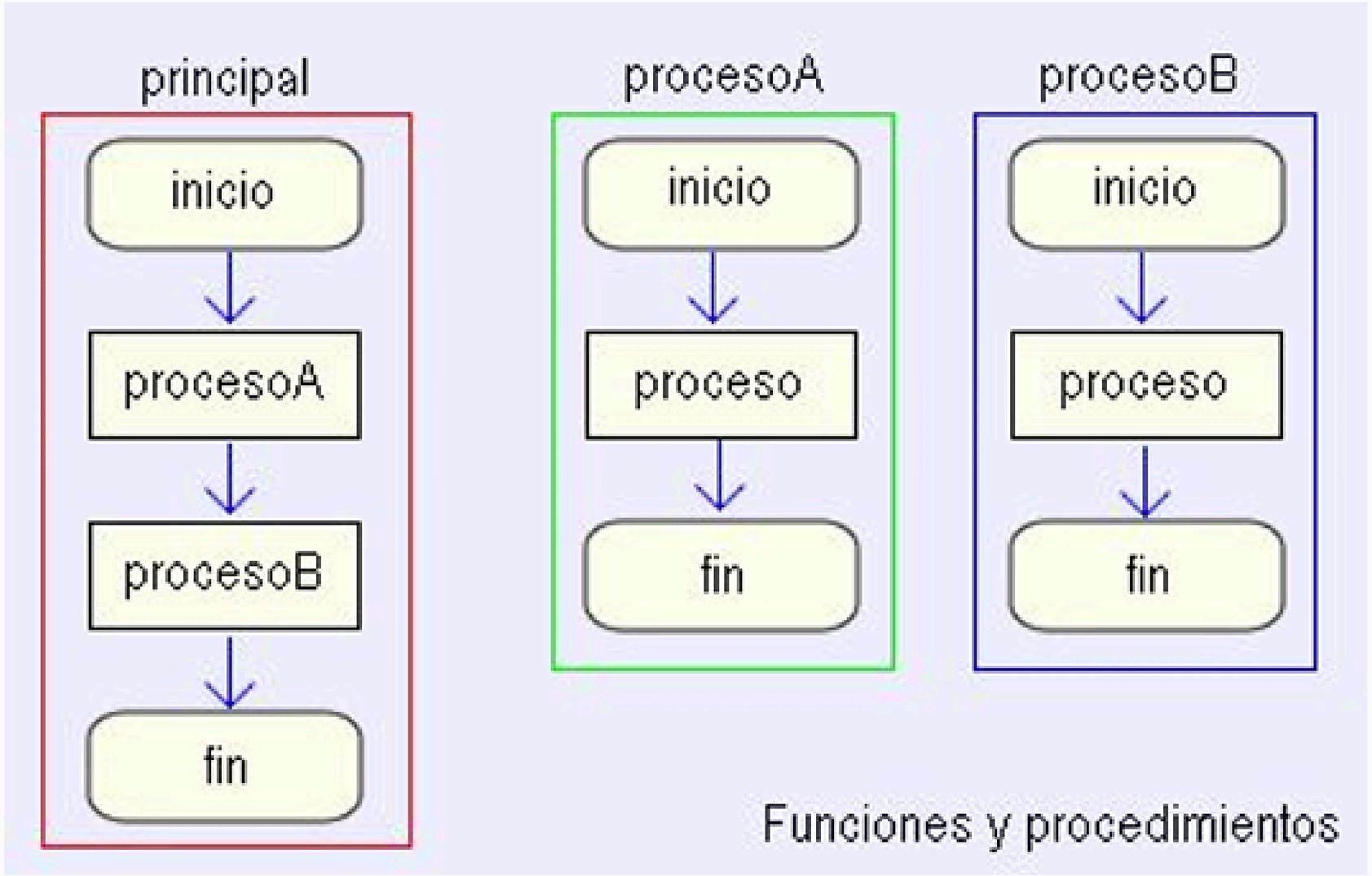
FUNCIONES Y METODOS

INTRODUCCIÓN

- La resolución de problemas complejos se facilita considerablemente si se dividen en problemas más pequeños; y la resolución de estos subproblemas se realiza mediante subalgoritmos.
- Los subalgoritmos son unidades de programa o módulos que están diseñados para ejecutar alguna tarea específica.
- Estos, son constituidos por funciones o procedimientos (métodos), los cuales se escriben solamente una vez, pero pueden ser referenciados en diferentes puntos del programa, de modo que se puede evitar la duplicación innecesaria del código.
- El módulo principal se ejecuta en una primera instancia, que da la orden de inicio de ejecución de los subprogramas.


principal
inicio
procesoA
procesoB
fin
procesoA
inicio
proceso
fin
procesoB
inicio
proceso
fin
Funciones y procedimientos

FUNCIONES DEFINIÓN

- Una función es un conjunto de líneas de código que realizan una tarea específica y retornan un valor.
- Las funciones pueden tomar parámetros que modifiquen su funcionamiento.
- Las funciones son utilizadas para descomponer grandes problemas en tareas simples y para implementar operaciones que son comúnmente utilizadas durante un programa y de esta manera reducir la cantidad de código.
- Cuando una función es invocada se le pasa el control a la misma, una vez que esta finalizó con su tarea el control es devuelto al punto desde el cual la función fue llamada.

```
<tipo> <nombre> ( [Parámetros] )
{
    cuerpo;
}
```

```
// regresar el cuadrado de un número
double Cuadrado(double n)
{
    return n*n;
}
```

METODOS DEFINIÓN

- Bajo ciertas circunstancias se necesitará escribir funciones que no regresen valor alguno y para ello se podrán declarar a la función como void.
- La palabra reservada void es utilizada para declarar funciones sin valor de retorno.
- Las funciones que retornan void no precisan de la palabra reservada return
- La principal diferencia entre una función y un método es que el primero debe retornar un valor y el segundo no.

```
<tipo> <nombre> ( [Parámetros] )
{
    cuerpo;
}
```

```
void  ImprimeCuadrado(double n)
{
   printstr( n*n );
}
```

PARAMETROS DEFINIÓN

- Las funciones y los métodos operan sobre ciertos valores que son pasados a los mismos ya sea como variables.
- Estas variables se denominan parámetros o argumentos.
- Existen lenguajes que permiten que los valores de estos parámetros sean pasados de dos maneras: por referencia o por valor.
- Los parámetros pasados por referencia pueden ser alterados por la función que los reciba, mientras que los parámetros pasados por valor o copia no pueden ser alterados por la función que los recibe, es decir, la función puede manipular a su antojo al parámetro, pero ningún cambio hecho sobre este se reflejará en el parámetro original.

EJEMPLO

1. Tenemos un método principal main.
2. Definimos una función (método en Java).
3. Llamamos a esa función desde el main.
4. Le pasamos valores por parámetros.

EJEMPLO

```
public class EjemploParametros {

    // Función (método) que recibe parámetros
    public static int sumar(int a, int b) {
        return a + b;
    }
    // Método principal
    public static void main(String[] args) {
        // Llamada a la función con parámetros
        int resultado = sumar(5, 3);
        // Mostrar el resultado
        System.out.println("La suma es: " + resultado);
    }
}
```

- public static int sumar(int a, int b) → es un método que recibe dos parámetros (a y b) y devuelve su suma.
- main llama a sumar(5, 3) y guarda el valor en resultado.
- Luego se imprime en pantalla.

PARAMETROS POR VALOR

Pasaje por valor:

- Es cuando a una función o método se le envía una copia del valor de la variable original.
- De esta manera, cualquier cambio que se haga dentro de la función afecta solo a la copia local y no modifica la variable original.
- En Java, esto ocurre con los tipos primitivos (int, double, char, etc.).

PARAMETROS POR VALOR

```
public class PasajePorValor {
    // Método que intenta cambiar el valor
    public static void add10(int n) {
        n = n + 10;  // se modifica la copia local de n
        System.out.println("Dentro de la función: n = " + n);
    }
    public static void main(String[] args) {
        int num = 10;
        System.out.println("Antes de llamar a la función: num = " + num);
        add10(num);
        System.out.println("Después de llamar a la función: num = " + num);
    }
}
```

PARAMETROS POR REFERENCIA

Pasaje por referencia:

- Es cuando a una función o método se le envía la dirección de memoria de la variable original, o una referencia al objeto.
- En este caso, los cambios realizados dentro de la función sí afectan a la variable u objeto original, porque ambas (la función y el programa principal) están trabajando sobre el mismo espacio en memoria.
- En Java no hay punteros como en C/C++, pero con los objetos se simula este comportamiento, ya que lo que se pasa es una copia de la referencia al objeto.

```
class Numero {
    int valor;
    Numero(int valor) {
        this.valor = valor;
    }
}
public class PasajePorReferencia {
    // Método que modifica el objeto
    public static void add10(Numero n) {
        n.valor = n.valor + 10;  // modifica el objeto real
        System.out.println("Dentro de la función: valor = " + n.valor);
    }
    public static void main(String[] args) {
        Numero num = new Numero(10);
        System.out.println("Antes de llamar a la función: valor = " + num.valor);
        add10(num);
        System.out.println("Después de llamar a la función: valor = " + num.valor);
    }
}
```

PARAMETROS POR REFERENCIA

Aclaración de Pasaje por Referencia (simulado en Java con objetos)

1. Cuando llamamos a add10(num), lo que se pasa es una copia de la referencia al objeto num.
2. Esa copia apunta al mismo lugar en memoria (ej: 1000) donde está el objeto.
3. Dentro de add10(), hacemos n.valor = n.valor + 10, por lo que realmente se modifica el mismo objeto.
4. Como ambas referencias apuntan al mismo objeto, el valor cambia tanto dentro como fuera de la función.

ALCANCES DE VARIABLES Y PARAMETROS

Tanto las variables como los parámetros de las funciones y métodos poseen diferentes alcances (scopes). Los mismos permiten determinar los valores de las mismas, ya se que éstas se definan con el mismo nombre en distintos scopes del programa.