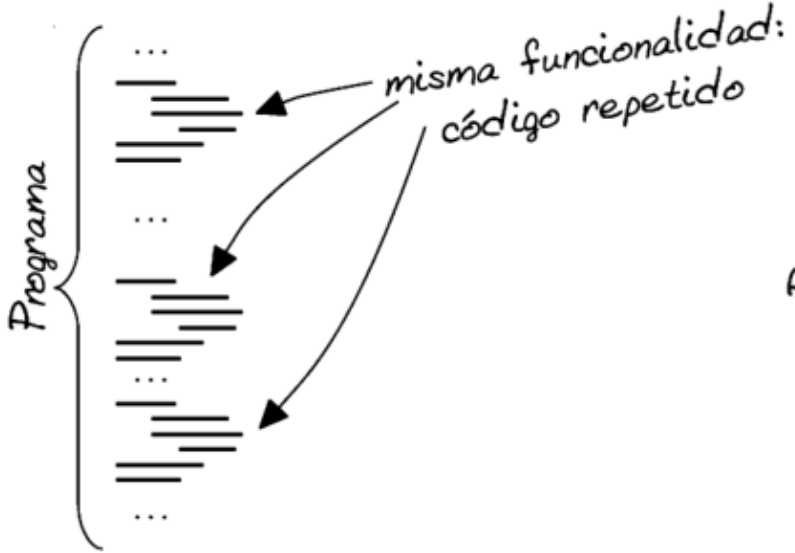
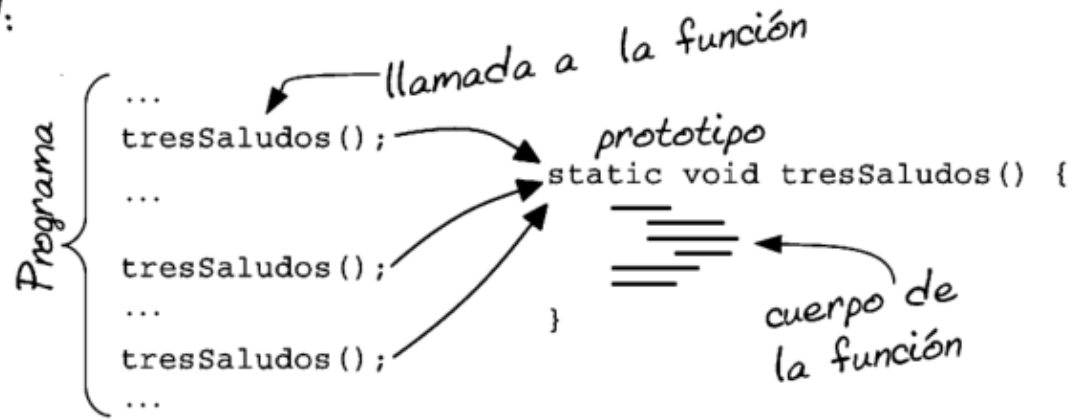
Funciones - utilidad





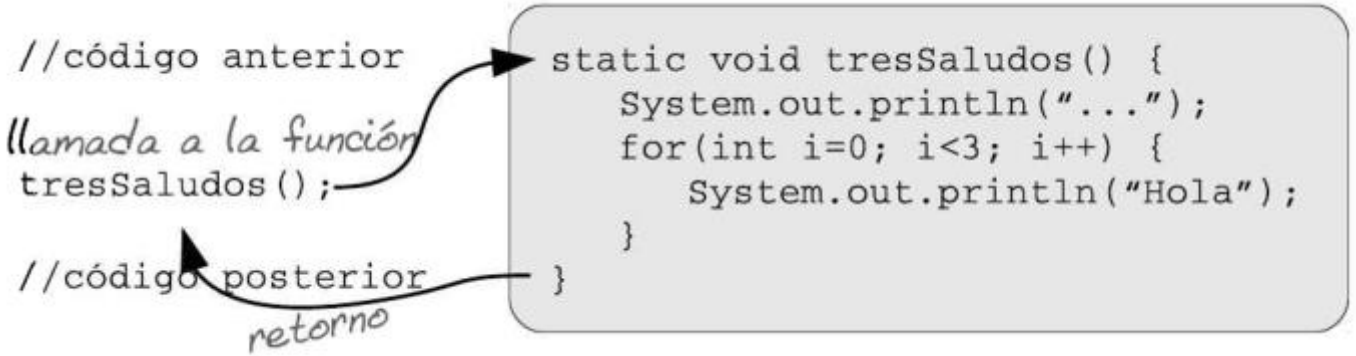
Funciones - declaración

```
static tipo nombreFunción() {
   cuerpo de la función
}
```

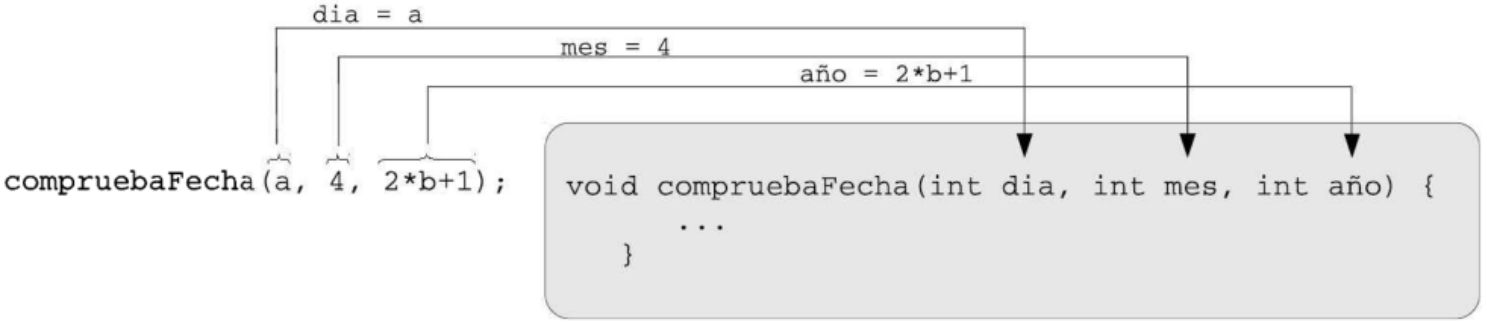
```
static tipo nombreFunción(tipo1 parametro1, tipo2 parametro2...) {
   cuerpo de la función
}
```

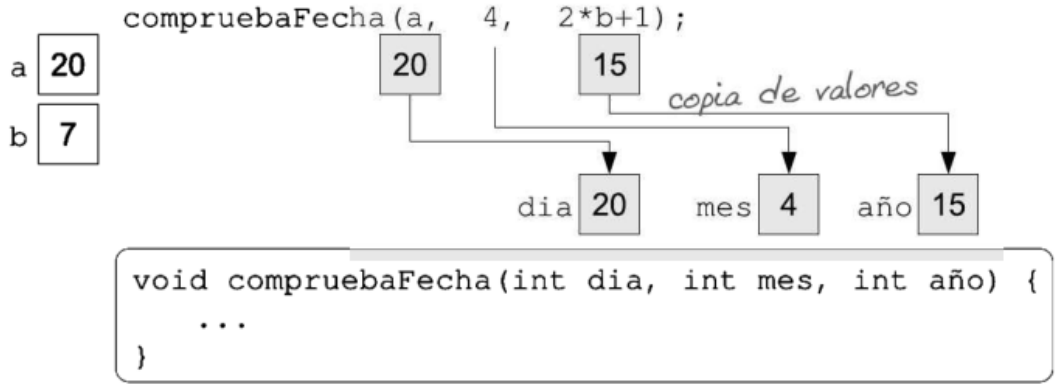
```
static int suma(int x, int y) {
   int resultado;
   resultado = x + y;

   return resultado;
}
```

Funciones - ejecución

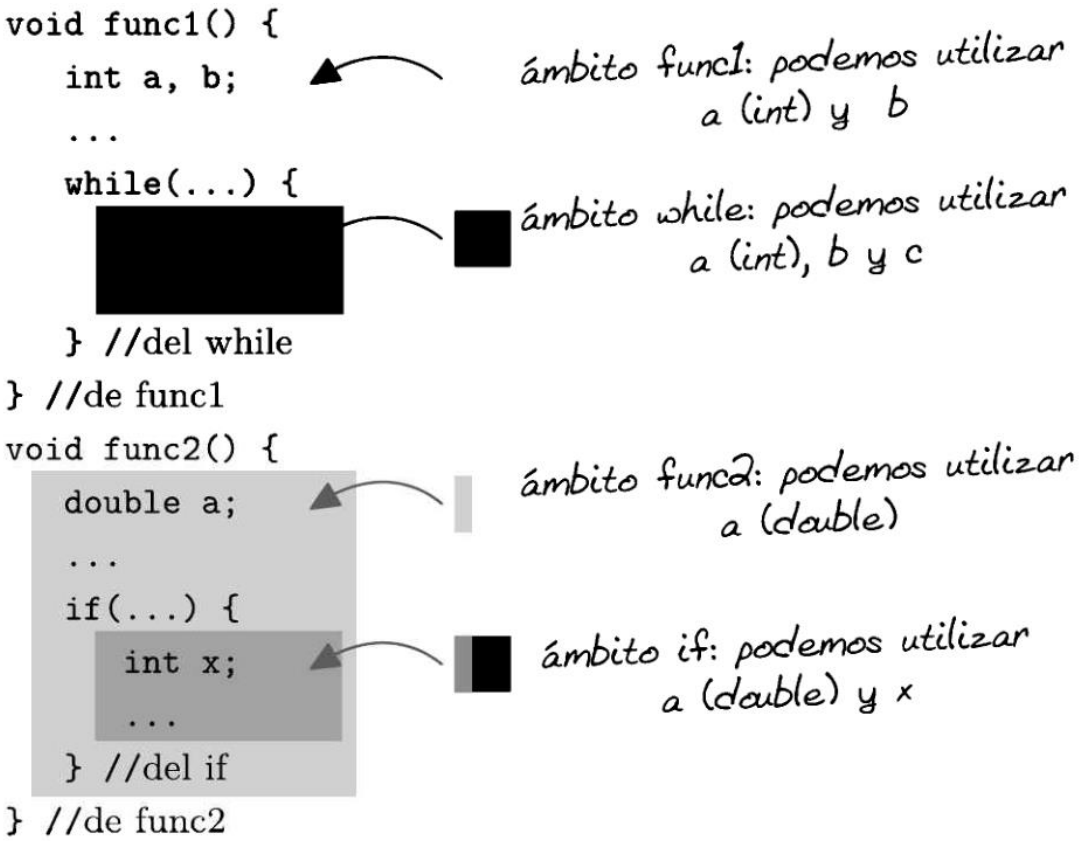


Funciones - orden de parámetros





Funciones - ámbito de variables



Funciones - sobrecarga de funciones

```
static int suma(int x, int y) {
 int resultado;

 resultado = x + y;

 return resultado;
}
```

```
static double suma(int a, double pesoA, int b, double pesoB) {
 double resultado;

 resultado = a * pesoA / (pesoA + pesoB) + b * pesoB / (pesoA + pesoB);

 return resultado;
}
```

OJO: Una función sobrecargada se distingue de otra por su número o tipo de parámetros, pero NO por el tipo que devuelve

Funciones - recursividad

Definición: Cuando una función se invoca a sí misma, se dice que es recursiva

```
static int funcionRecursiva() {
   //...
   funcionRecursiva();  // llamada recursiva
   //...
}
```

```
static int funcionRecursiva(int datos) {

 int resultado;

 if (caso_base) {
   resultado = valorBase;
 } else {
   resultado = funcionRecursiva(nuevos_datos); // llamada recursiva
 }

 return resultado;
}
```

Funciones - recursividad (ejemplo)

```
static int factorial(int n) {
 int resultado;

 if (n == 0) {
   resultado = 1; // caso base
 } else {
   resultado = n * factorial(n - 1); // llamada recursiva
 }

 return resultado;
}
```