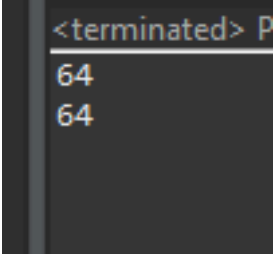


Éste será un concepto sencillo, el **Paso de Valores por Referencia** consiste en cambiar valores de una variable, pasándosela como parámetro a una función, quedaos con la anterior frase, y en un rato la entenderéis:

```
public class PasoDeValor {  
    public static void fun (int num) {  
        num = 5;  
    }  
  
    public static void main (String[] args) {  
        int a = 64;  
        System.out.println(a);  
        fun(a);  
        System.out.println(a);  
    }  
}
```

>>>



Como veis creo el entero *a* y lo inicializo en 64, y luego al llamar a la función *fun*, a pesar de que cambie el valor del parámetro, no cambia el valor de la variable de **referencia**, esto ocurre con cualquier cosa: *ints*, *booleans*, *doubles*, *chars*...

Menos con los *arrays*, al cambiar el valor de un elemento de un *array* que se toma como parámetro en una función, cambia tanto el valor del parámetro como el de la variable original:

```
public class PasoDeValor {  
    public static void fun (int[] col) {  
        col[2] = 5;  
    }  
  
    public static void main (String[] args) {  
        int[] nums = {100,200,300,400,500};  
        System.out.println(nums[2]);  
        fun(nums);  
        System.out.println(nums[2]);  
    }  
}
```

>>>



Ahora creo un *array* de *ints* con 5 elementos, y en la función cambio el elemento con índice 2 por un 5, y luego como veís al volverlo a imprimir se ha cambiado dicho valor, esto solo ocurre con *arrays*, y bajo una condición que dire después de la siguiente aclaración; no es necesario que el parámetro y la variable de origen tengan el mismo nombre, y aunque parezca un poco obvio, por eso lo llamo aclaración, ya que algunas personas pueden creer que es necesario que el parámetro y la variable tengan el mismo nombre, es decir, que en vez de llamarse *nums* se tendría que llamar *col* también o viciersa, dicho esto veamos un concepto muy importante.

No podemos cambiar de tamaño al *array* mientras esté dentro de la función, osea, no podemos volver a inicializarlo, si nuestra función recibe como parámetro el *array* *col*, y queremos modificar el *array* original no podemos escribir **NUNCA** esto:

- `col = new int [longitud];`

Incluso aunque tengan el mismo tamaño que el original, da igual, si ponemos →
`col = new int [col.length];`

Deja de ser Paso por Referencia, ya que el objeto por referencia que hemos tomado a sido borrado, esto es algo que veremos más adelante por qué sucede, además os diré que no podéis escribir $col = \{X_0, X_1, \dots, X_n\}$, ya que esto de crear *arrays* por enumeración solo se puede hacer al crearlos, es decir, cuando ponemos `int[] col = {X0, X1, ..., Xn}`, eso sí estaría bien, recordad que crear una variable es decir que existe, e inicializarla es asignarle un valor, en *arrays* solo podemos inicializar por enumeración si lo hacemos a la vez que lo creamos, y no sólo eso, en una función no podemos crear una variable que existe como parámetro, por lo que la única forma de fastidiar el paso de valor por referencia sería poniendo lo que dije antes → `col = new int [longitud]`, veamos un ejemplo:

```
public class PasoDeValor {  
    public static void fun (int[] col) {  
        col = new int[5];  
        col[2] = 5;  
    }  
    public static void main (String[] args) {  
        int[] nums = {100,200,300,400,500};  
        System.out.println(nums[2]);  
        fun(nums);  
        System.out.println(nums[2]);  
    }  
}
```

>>>

```
<terminated> PasoDeValor  
300  
300
```

Solo es eso, dejaré un único ejercicio especializado en paso por referencia dentro de los ejercicios de *arrays* en mi Drive.