

Parámetros en Java

Java no tiene pasaje de parámetros por referencia. Los parámetros se pasan por valor.

Java no pasa objetos como parámetros, sino copias de las referencias a esos objetos.

El pasaje por referencia, y el pasaje de una referencia son dos cosas diferentes.

En Java no hay pasaje por referencia. El modificador var de Pascal no existe en JAVA.

Ejemplos Incorrectos

Supongamos que se quiere calcular e imprimir el máximo valor de un arreglo.

```
public class Maximo {  
    private void buscarMaximo(int[] a, int max) {  
        for (int i=0;i<a.length;++i)  
            if (a[i] > max) max=a[i];  
    }  
    public void imprimirMaximo(int[] a) {  
        int max = 0;  
        //el valor de "max" nunca se modifica  
        buscarMaximo(a, max);  
        // Siempre imprime 0  
        System.out.print(max);  
    }  
}
```

max es de tipo primitivo, lo que se pasa ahí es una copia del valor de max, es decir 0. Es lo mismo que hacer: `buscarMaximo(a, 0);`

```
public class Maximo {  
    private void buscarMaximo(int[] a, Integer max) {  
        for (int i=0;i<a.length;++i)  
            if (a[i] > max) max=a[i];  
    }  
    public void imprimirMaximo(int[] a) {  
        Integer max = 0;  
        //el valor de "max" nunca se modifica  
        buscarMaximo(a, max);  
        // Siempre imprime 0.  
        System.out.print(max);  
    }  
}
```

Tampoco funciona si pasamos un **Integer max**. Los objetos **Integer** (y todos los *wrappers*) son inmutables !!!.


Un primer ejemplo correcto

Una solución de usar la sentencia **return** para devolver un valor. Esta es ideal pero no siempre es posible

```
public class Maximo {  
    public int buscarMaximo(int[] a) {  
        int max = 0; // max es local  
        for (int i=0; i<a.length; ++i)  
            if (a[i] > max) max = a[i];  
        return max;  
    }  
    private void imprimirMaximo(int[] a) {  
        int max = buscarMaximo(a);  
        System.out.print(max);  
    }  
}
```

Otra alternativa

Otra solución podría ser usar una variable de instancia para mantener la suma. OJO: esta solución podría llegar a ensuciar la estructura cuando se pide que implementen los algoritmos en los árboles, grafos, etc.

```
public class Maximo {  
    private int max;  No abusar de esta  
                        opción.  
  
    void buscarMaximo(int[] a) {  
        max = 0; // variable de instancia  
        for (int i=0; i<a.length; ++i)  
            if (a[i] > max) {  
                max = a[i];  
            }  
    }  
  
    void imprimirMaximo(int[] a) {  
        buscarMaximo(a);  
        System.out.print(max);  
    }  
}
```

Otros ejemplos correctos

Otra solución podría ser usar un *objeto simple* para sumar


Usar un arreglo con un elemento para sumar.

En este caso, lo que recibe `buscarmaximo(a, max)` en `max` es una copia de la referencia al arreglo. La referencia original y la recibida por parámetro son iguales, apuntan al mismo objeto. De esta manera el valor se actualiza.

```
public class Maximo {  
    void buscarMaximo(int[] a, int[] max) {  
        for (int i=0;i<a.length;++i)  
            if (a[i]>max[0]) max[0]=a[i];  
    }  
    void imprimirMaximo(int[] a) {  
        int[] max = {0};  
        buscarMaximo(a, max);  
        System.out.print(max[0]);  
    }  
}
```

Una solución más genérica podría ser definir una clase simple con el o los contadores

```
public class Maximo {  
    void buscarMaximo(int[] a, MaxUtil max) {  
        for (int i=0;i<a.length;++i)  
            if (a[i]>max.getC1()) max.setC1(a[i]);  
    }  
    void imprimirMaximo(int[] a) {  
        MaxUtil max = new MaxUtil();  
        buscarMaximo(a, max);  
        System.out.print(max.getC1());  
    }  
}
```



```
public class MaxUtil {  
    private int c1;  
    public int getC1() {  
        return c1;  
    }  
    public void setC1(int c1) {  
        this.c1 = c1;  
    }  
}
```