

# Pasaje de parámetros en Java

En Java los parámetros se pasan por valor. Pasaje por valor significa que cuando se invoca a un método, se pasan como argumentos al método una copia de cada parámetro actual.

```
package ayed.tp02;
```

```
public class PasajePorValor {  
    public static int mult(int x, int y) {  
        return x * y;  
    }  
    public static void main(String[] args) {  
        int alto = 10;  
        int ancho = 5;  
        int area = mult(alto, ancho);  
    }  
}
```

**Parámetro formal:**  
parámetros en la  
definición de un método.

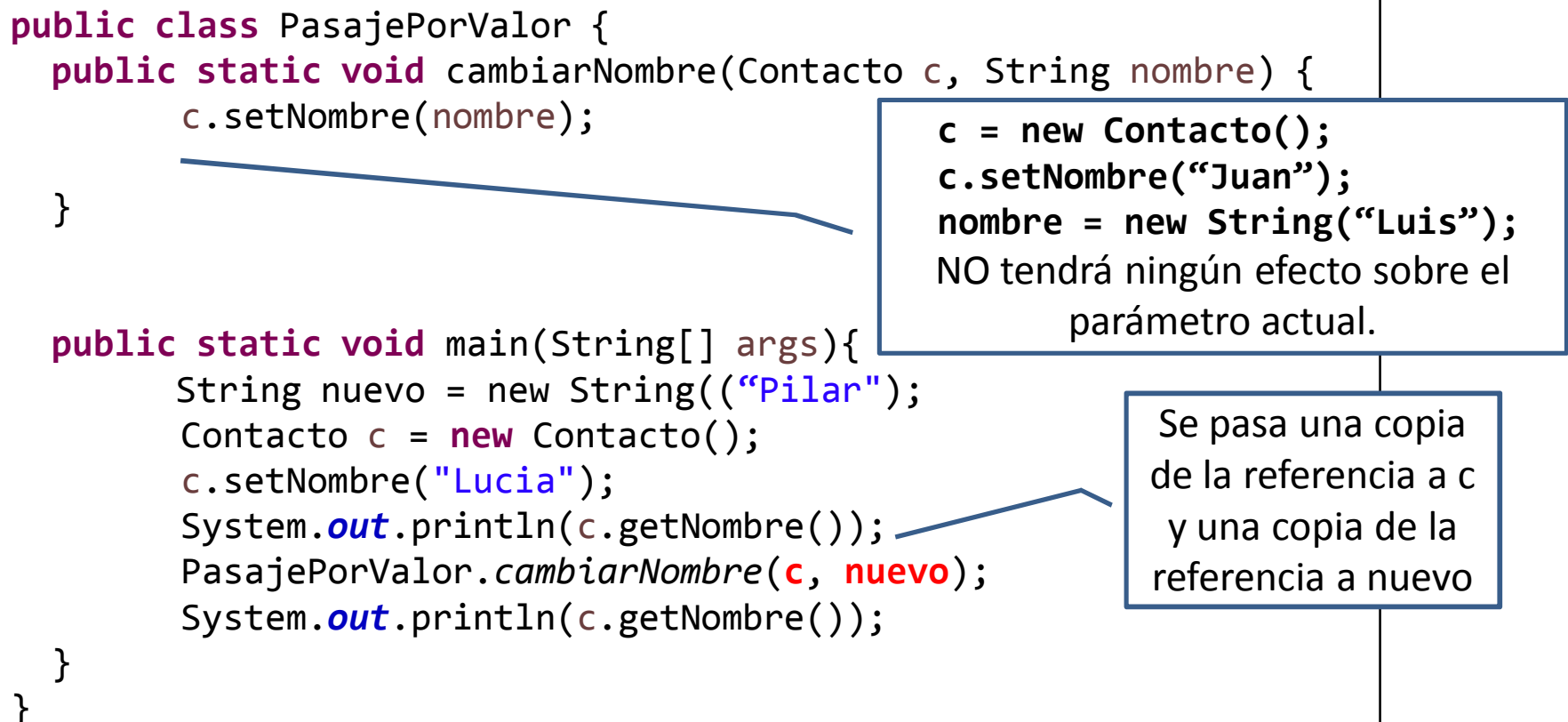
**Parámetro actual:**  
parámetros en la invocación  
del método.

Adentro del método se puede cambiar el valor de esa copia, pero ésto no tendrá efecto en el parámetro actual.

# Pasaje de parámetros en Java

”Adentro del método se puede cambiar el valor de esa copia, pero ésto no tendrá efecto en el parámetro actual”

**¿Es un poco restrictivo esto?** No. En Java, podemos pasar como parámetro una referencia a un objeto y con ella cambiar algo adentro de ese objeto, pero no podemos cambiar a qué objeto se refiere.



# Calcular el máximo de un arreglo

Dado un arreglo con valores de tipo `int` se desea calcular el máximo valor del arreglo

```
package tp02.ejercicio5;

public class Test {

    public static void main(String[] args) {
        int[] datos = {3, 4, 5, 8, 0};
        int max = Calculadora.maximo(datos);
        System.out.println("El máximo es " + max);
    }
}
```

- ¿Cómo implementamos el método `maximo(arreglo)`?
- ¿Es un método de instancia o de clase?
- ¿Qué parámetros le pasamos?

# Calcular el máximo de un arreglo

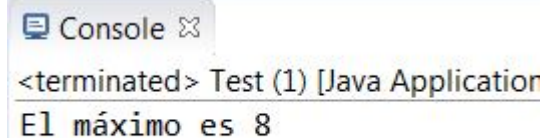
La manera más simple es usar la sentencia **return** para devolver el máximo del arreglo.

```
package ayed.tp02;

public class Calculadora {
    public static int maximo(int[] datos) {
        int max = 0;
        for (int i = 0; i < datos.length; i++) {
            if (datos[i] > max)
                max = datos[i];
        }
        return max;
    }
}

package ayed.tp02;

public class Test {
    public static void main(String[] args) {
        int[] datos = {3, 4, 5, 8, 0};
        int max = Calculadora.maximo(datos);
        System.out.println("El máximo es " + max);
    }
}
```



Console

<terminated> Test (1) [Java Application]

El máximo es 8

¿Cómo podríamos hacer si tenemos que devolver el máximo y el mínimo?

# Calcular el máximo y el mínimo de un arreglo

De la misma manera que devolvemos un arreglo, podemos devolver un objeto con el máximo y el mínimo.

```
package ayed.tp02;

public class Calculadora {
    . . .

    public static Datos maxmin(int[] datos) {
        int max = 0, min = 0;
        for (int i = 0; i < datos.length; i++) {
            if (datos[i] > max) max = datos[i];
            if (datos[i] < min) min=datos[i];
        }
        Datos obj = new Datos();
        obj.setMax(max);
        obj.setMin(min);
        return obj;
    }
}
```

```
package ayed.tp02;

public class Datos {
    private int min;
    private int max;
    public int getMin() {
        return min;
    }
    public void setMin(int min) {
        this.min = min;
    }
    public int getMax() {
        return max;
    }
    public void setMax(int max) {
        this.max = max;
    }
}
```

En este caso creamos una objeto que mantiene el máximo y el mínimo y devolvemos ese objeto.

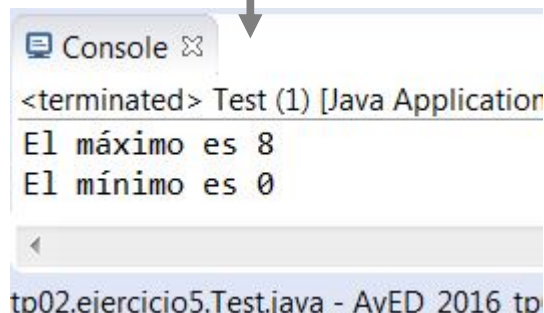
# Calcular el mínimo y el máximo de un arreglo

Ahora al invocar al método nos devuelve un objeto:

```
package tp02.ejercicio5;

public class Test {

    public static void main(String[] args) {
        int[] datos = {3, 4, 5, 8, 0};
        Datos maxmin = Calculadora.maxmin(datos);
        System.out.println("El máximo es " + maxmin.getMax());
        System.out.println("El mínimo es " + maxmin.getMin());
    }
}
```



Console

<terminated> Test (1) [Java Application]

El máximo es 8

El mínimo es 0

tp02.ejercicio5.Test.java - AyED 2016 tp

Continuemos analizando otras alternativas . . .

# Otra alternativa para calcular el máximo de un arreglo

Otra alternativa es guardar valores de cálculos parciales o totales en variables de instancia o clase.

```
package tp02.ejercicio5;

public class Calculadora {
    private static int max;
    private static int min;

    public static void buscarMaxMin(int[] a){
        for (int i = 0; i < a.length; ++i)
            if (a[i] > max) {
                max = a[i];
            }
            if (a[i] < min)
                min = a[i];
        }
    }

    public static int getMax() {
        return max;
    }

    public static int getMin() {
        return min;
    }
}
```

```
public class Test {
    public static void main(String[] args) {
        int[] datos = { 3, 4, 5, 8, 0 };
        Calculadora.buscarMaxMin(datos);
        System.out.println("El máximo es " +
                           Calculadora.getMax());
        System.out.println("El mínimo es " +
                           Calculadora.getMin());
    }
}
```

# Otro ejemplo con varios tipos

The screenshot shows an IDE with two Java files open. The top file, `Modificador.java`, contains a class `Modificador` with a static method `modificar` that takes various parameters and modifies them. The bottom file, `ParametrosTest.java`, contains a class `ParametrosTest` with a `main` method that creates variables of different types, calls `Modificador.modificar`, and prints the results. The console window shows the output of the `main` method.

```
1 package pruebas;
2
3 public class Modificador {
4     public static void modificar(int x, Integer y, String palabra, int[] arre, Contacto c1, Contacto c2) {
5         x = x + 20;
6         y = y + 20;
7         palabra = palabra + "Juana";
8         arre[0] = arre[0] + 30;
9         c1 = new Contacto();
10        c1.setNombre("Juana");
11        c2.setNombre("Juana");
12    }
13 }
14 }
```

```
1 package pruebas;
2
3 public class ParametrosTest {
4     public static void main(String[] args) {
5         int a = 10;
6         Integer b = new Integer(20);
7         String str = new String("Hola");
8         int[] arre = { 10 };
9         System.out.println("a: " + a + " b: " + b + " str: " + str);
10        Contacto c1 = new Contacto();
11        c1.setNombre("Pedro");
12        Contacto c2 = new Contacto();
13        c2.setNombre("Pedro");
14        Modificador.modificar(a, b, str, arre, c1, c2);
15        System.out.println("a: " + a + " b: " + b + " str: " + str);
16        System.out.println("nombre de c1: " + c1.getNombre() + " nombre de c2: " + c2.getNombre());
17    }
18 }
19 }
```

Problems @ Javadoc Declaration Console

<terminated> ParametrosTest [Java Application] C:\Prograr

a: 10 b: 20 str: Hola  
a: 10 b: 20 str: Hola  
nombre de c1: Pedro nombre de c2: Juana

Las clases Wrapper y  
los String son inmutables!!