GUÍA VISUAL ARRAYS

```
int[] datos=new int[5];
```

0	1	2	3	4

```
datos[0]=25;
datos[1]=30;
```

0	1	2	3	4
25	30			

```
int[] numeros = new int[5] {200, 150, 100, -50, 300};
```

0	1	2	3	4
200	150	100	-50	300

Operaciones habituales con arrays:añadir, insertar, borrar

añadir un dato al final de los ya existentes (array sobredimensionado)

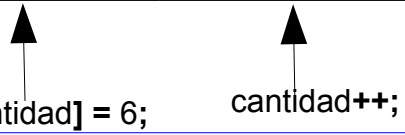
```
int[] datos = {10, 15, 12, 0, 0};
int capacidad = 5; // Capacidad máxima del array
int cantidad = 3; // Número real de datos guardados, que coincide con la primera posición libre.
```

0	1	2	3	4
10	15	12	0	0

cantidad=3
Primera posición libre

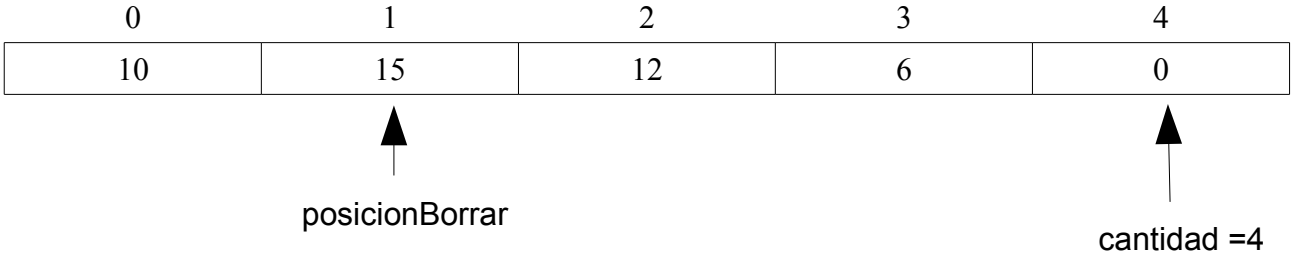
```
// Añadimos un dato al final
Console.WriteLine("Añadiendo 6 al final");
if (cantidad < capacidad)
{
        datos[cantidad] = 6;
        cantidad++;
}
```

0	1	2	3	4
10	15	12	6	0

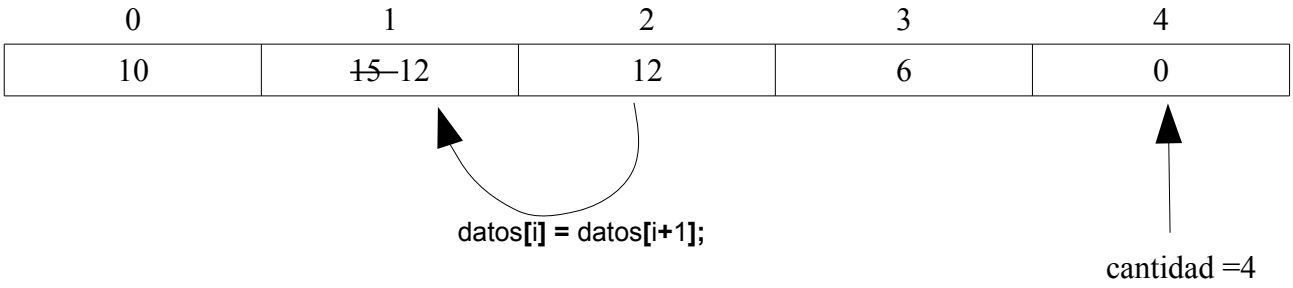


datos[cantidad] = 6; cantidad++;

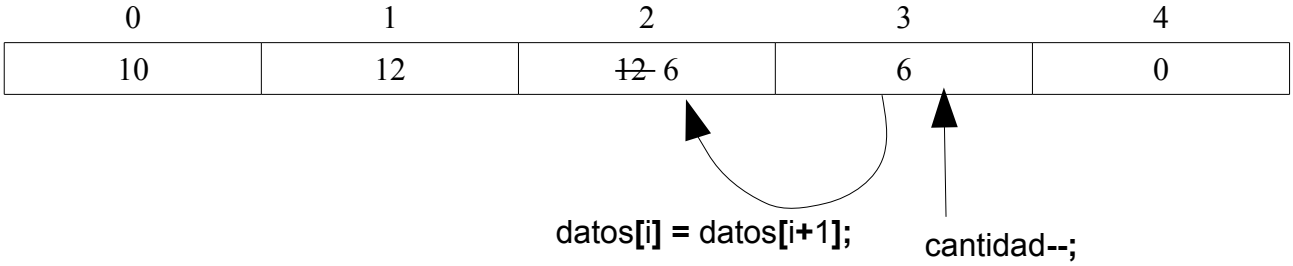
```
// Borramos el segundo dato
Console.WriteLine("Borrando el segundo dato");
int posicionBorrar = 1;
for (i=posicionBorrar; i<cantidad-1; i++)
        datos[i] = datos[i+1];
cantidad--;
```



Primera iteración (para i = 1)

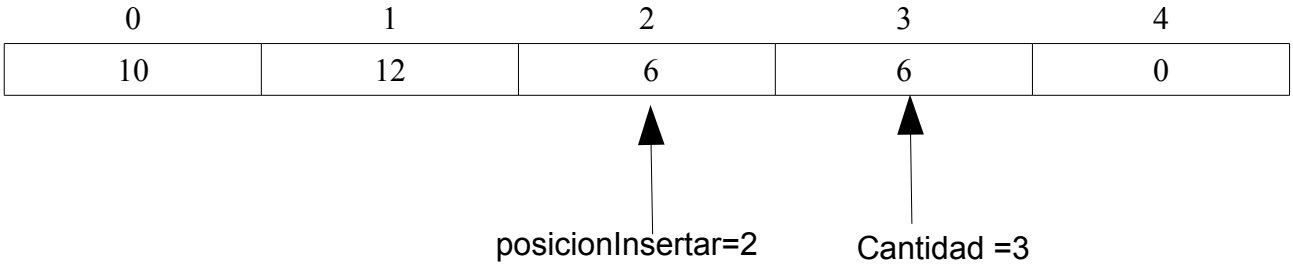


(para i = 2)

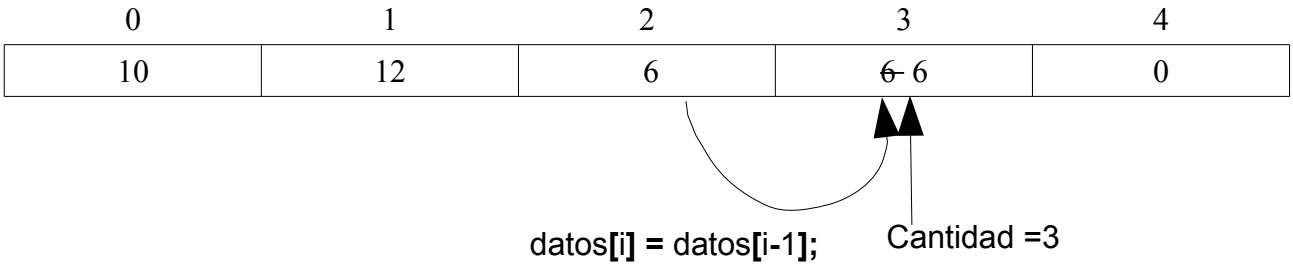


Terminamos el bucle ya que en la siguiente iteración i sería 3 y deja de cumplir la condición del for.
Aunque en la posición 3 siga habiendo un 6, como el valor de cantidad es 3, significa que esa posición está libre para ser utilizada, no importa si tiene un 6 o un 0. No es necesario borrarla en sí, simplemente la hemos marcado como disponible.

```
// Insertamos 30 en la tercera posición
if (cantidad < capacidad)
{
Console.WriteLine("Insertando 30 en la posición 3");
int posicionInsertar = 2;
for (i=cantidad; i>posicionInsertar; i--)
datos[i] = datos[i-1];
datos[posicionInsertar] = 30;
cantidad++;
}
```

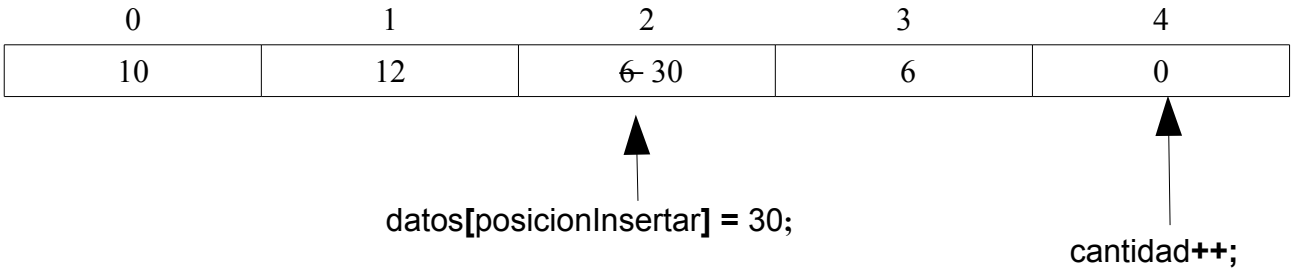


(para i=3)



El for no realiza más iteraciones.

```
datos[posicionInsertar] = 30;
cantidad++;
```



Arrays bidimensionales

```
int[,] notas1 = new int[2,2]; // 2 bloques de 2 datos
notas1[0,0] = 1;
notas1[0,1] = 2;
notas1[1,0] = 3;
notas1[1,1] = 4;
```

	0	1
0	1	2
1	3	4

```
int[,] notas2 = // 2 bloques de 10 datos, prefijados
{
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10},
{11, 12, 13, 14, 15, 16, 17, 18, 19, 20}
};
```

	0	1	2	3	4	5	6	7	8	9
0	1	2	3	4	5	6	7	8	9	10
1	11	12	13	14	15	16	17	18	19	20

Arrays multidimensionales

```
int[][] notas; // Array de dos dimensiones
notas = new int[3][]; // Serán 3 bloques de datos
notas[0] = new int[10]; // 10 notas en un grupo
notas[1] = new int[15]; // 15 notas en otro grupo
notas[2] = new int[12]; // 12 notas en el último
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0															
1															
2															

```
// Damos valores de ejemplo
for (int i=0;i<notas.Length;i++)
{
for (int j=0;j<notas[i].Length;j++)
{
notas[i][j] = i + j;
}
}
```