



UNIVERSIDAD DE ALMERÍA

Grado en Ingeniería Informática

Introducción a la Programación

2016-2017



Tema 4. Arrays

- Arrays unidimensionales
- Arrays multidimensionales

Tema 4. Arrays

Arrays unidimensionales

- ☐ Introducción
- ☐ Declaración y creación de un array
- ☐ Acceso a los elementos de un array
- ☐ Procesamiento en un array
- ☐ La clase Arrays

Tema 4. Arrays

Arrays unidimensionales

Introducción

Supongamos que necesitamos leer 100 números, calcular su media y encontrar cuantos de ellos tienen un valor superior a la media. Nuestro programa debería en primer lugar leer los números y calcular a continuación la media para después comparar ese valor con cada uno de ellos y determinar si es superior o no. Los números se deben almacenar en variables, ello supondría la declaración de 100 variables y el programa resultaría poco práctico. ¿Cómo solucionar el problema?.

Java y los lenguajes de alto nivel proporcionan una estructura de datos, *arrays*, que permiten dar una solución más eficiente al problema. En lugar de declarar 100 variables individuales tales como `numero0`, `numero1`, ... y `numero99` declaramos una única variable array, `numeros`, y utilizamos `numeros[0]`, `numeros[1]` ... `numeros[99]` para representar elementos individuales.

Tema 4. Arrays

Arrays unidimensionales

Es la estructura de datos representativa del **acceso directo**. Es una colección de elementos del mismo tipo, es decir, es una **estructura de datos homogénea**, a la que podemos **acceder** a sus elementos individuales **a partir de un índice (posición)**. Es **una estructura de datos estática** y en cuanto al almacenamiento interno, los elementos del array **se almacenan en posiciones contiguas de memoria**.

Propiedades de los arrays

✓ Los arrays se utilizan como contenedores para almacenar datos relacionados (en vez de declarar variables por separado para cada uno de los elementos del array).

Tema 4. Arrays

Arrays unidimensionales

- ✓ Todos los **datos** incluidos en el array son **del mismo tipo**. Se pueden crear arrays de enteros de tipo `int` o de reales de tipo `float`, pero en un mismo array no se pueden mezclar datos de tipo `int` y datos de tipo `float`.
- ✓ El tamaño del array se establece cuando se crea el array (con el operador **`new`**, igual que cualquier otro objeto).
- ✓ A los elementos del array se accederá a través de la posición que ocupan dentro del conjunto de elementos del array.

Tema 4. Arrays

Arrays unidimensionales

☐ Declaración

Para declarar un array, se utilizan corchetes []

```
tipo identificador [];
```

o bien

```
tipo [] identificador;
```

donde

tipo es el tipo de dato de los elementos del array

identificador es el nombre de la variable array

Tema 4. Arrays

Arrays unidimensionales

☐ Creación

Los arrays se crean con el operador **new**

```
identificador = new tipo[elementos];
```

Entre corchetes se indica el tamaño del array, es decir, el número de elementos.

☐ Declaración y creación

```
tipo [] identificador = new tipo[elementos];
```


Tema 4. Arrays

Arrays unidimensionales

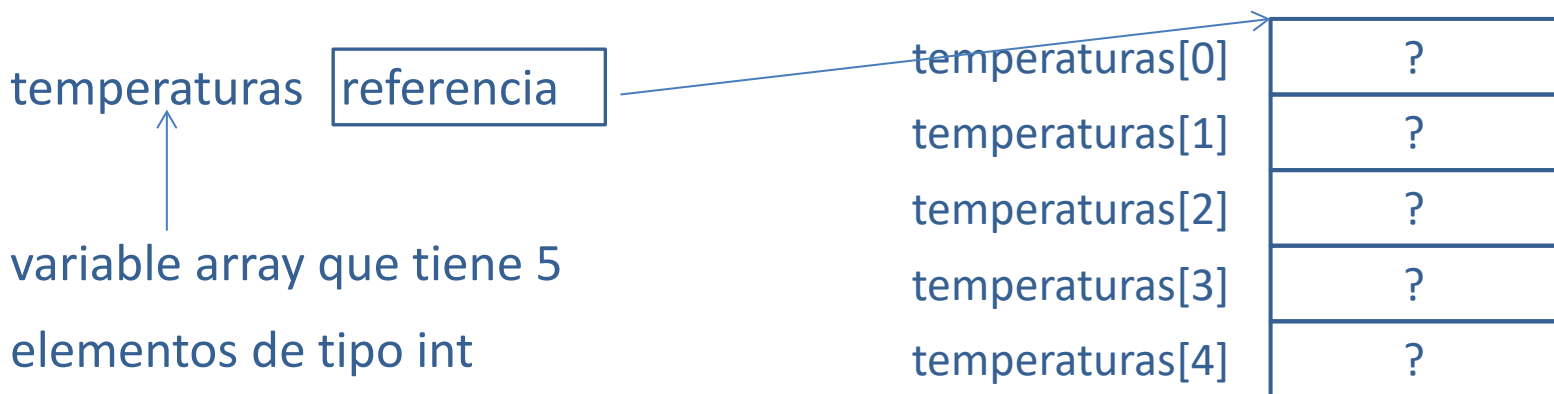
□ Ejemplos

```
final int NUMERO_DE_ELEMENTOS = 50;
```

```
float [] notas = new float[NUMERO_DE_ELEMENTOS];
```

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

```
Fraccion [] fracciones = new Fraccion[20];           // array de objetos
```



Tema 4. Arrays

Arrays unidimensionales

☐ Declaración, creación e inicialización

`tipo [] identificador = { valor0, valor1, ..., valork }`

Ejemplo

```
int [] temperaturas = {20, 35, 15, 10, -5};
```

temperaturas[0]

temperaturas[1]

temperaturas[2]

temperaturas[3]

temperaturas[4]

20
35
15
10
-5

Tema 4. Arrays

Arrays unidimensionales

- ❑ Declaración, creación e inicialización de un array de objetos

Ejemplo

```
Fraccion [] fracciones;
```

Declarar

```
fracciones = new Fraccion[3];
```

Crear

```
fracciones[0] = new Fraccion(8, 9);
```

```
fracciones[1] = new Fraccion(11, -22);
```

```
fracciones[2] = new Fraccion(1, 3);
```

Inicializar

En una sola línea:

```
Fraccion [] fracciones = {new Fraccion(8, 9), new Fraccion(11, -22), new  
Fraccion(1, 3)};
```

Tema 4. Arrays

Arrays unidimensionales

☐ Acceso a los elementos de un array

Para acceder a los elementos de un array utilizamos índices que indican la posición del elemento en el array.

temperaturas[índice]

- ✓ En Java, el **índice** del primer elemento de un array es siempre **0**.
- ✓ El tamaño del array puede obtenerse utilizando la propiedad **temperatura.length**
- ✓ Por tanto, el índice del último elemento es **temperatura.length-1**

Tema 4. Arrays

Arrays unidimensionales

Procesamiento en un array

Las operaciones se realizan elemento a elemento. Cuando realizamos una operación que afecta a todos o parte de los elementos de un array se utiliza un bucle **for** por dos razones:

- Todos los elementos del array son del mismo tipo, por tanto se procesarán del mismo modo usando un bucle **for**.
- El tamaño del array es conocido, resulta natural utilizar un bucle **for**.

Ejemplo 1. Almacenar en el array temperaturas, las temperaturas leídas de teclado.

```
Scanner entrada = new Scanner (System.in);  
System.out.println("Introduce los valores de las temperaturas");  
for (int i = 0; i < temperaturas.length; i++) {  
    System.out.println("Introduce temperatura " + (i + 1));  
    temperaturas[i] = entrada.nextInt();  
}
```

Tema 4. Arrays

Arrays unidimensionales

Ejemplo 2. Inicializar el array con valores aleatorios.

```
int N = 50;
for (int i = 0; i < temperaturas.length; i++) {
    temperaturas[i] = (int) (Math.random() * N);
}
```

Ejemplo 3. Mostrar el array.

```
for (int i = 0; i < temperaturas.length; i++) {
    System.out.print(temperaturas[i] + "\t");
}
```

Tema 4. Arrays

Arrays unidimensionales

Ejemplo 4. Calcular la media de todos los elementos.

```
double suma = 0.0, media;  
for (int i = 0; i < temperaturas.length; i++) {  
    suma += temperaturas[i] ;  
}  
media = suma / temperaturas.length;
```

Ejemplo 5. Encontrar el elemento mayor.

```
int max = temperaturas[0];  
for (int i = 1; i < temperaturas.length; i++) {  
    if (temperaturas[i] > max)  
        max = temperaturas[i];  
}
```

Tema 4. Arrays

Arrays unidimensionales

Ejemplo 6. Calcular la media de todos los elementos usando un método de clase (**static**).

```
public static double media(int [] temperaturas) {  
    double suma = 0.0;  
    for (int i = 0; i < temperaturas.length; i++) {  
        suma += temperaturas[i] ;  
    }  
    return suma / temperaturas.length;  
}
```


Tema 4. Arrays

Arrays unidimensionales

Ejemplo 7. Encontrar el menor índice del elemento con mayor valor.

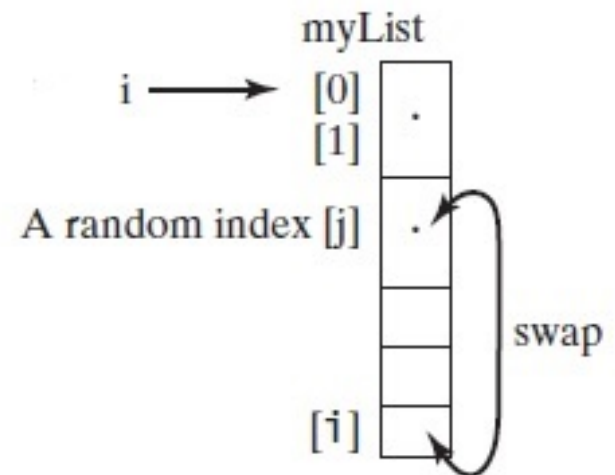
```
double max = temperaturas[0];  
int indiceMaxValor = 0;  
for (int i = 1; i < temperaturas.length; i++) {  
    if (temperaturas[i] > max) {  
        max = temperaturas[i] ;  
        indiceMaxValor = i;  
    }  
}
```

Tema 4. Arrays

➡ Arrays unidimensionales

Ejemplo 8. Aleatorizar el contenido del array (random suffling).

```
for (int i = temperaturas.length - 1; i > 0; i--) {  
    int j = (int)(Math.random() * (i + 1));  
    double temp = temperaturas[i] ;  
    temperaturas[i] = temperaturas[j];  
    temperaturas[j] = temp;  
}
```

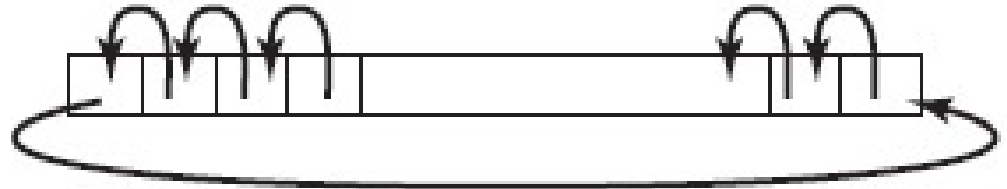


Tema 4. Arrays

➡ Arrays unidimensionales

Ejemplo 9. Desplazamiento de elementos en el array (Shifting elements).

```
double temp = temperaturas[0] ;  
for (int i = 1; i < temperaturas.length; i++) {  
    temperaturas[i - 1] = temperaturas[i];  
}  
temperaturas[temperaturas.length - 1] = temp;
```

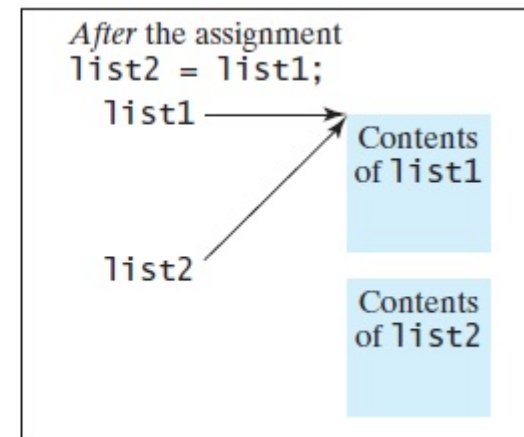
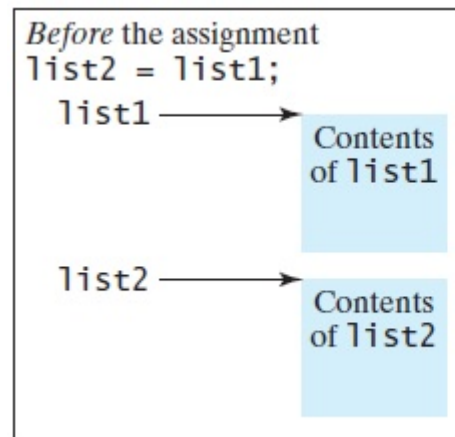


Tema 4. Arrays

➡ Arrays unidimensionales

Ejemplo 10. Copia de arrays (copiar el contenido de un array en otro, para ello tenemos que copiar elemento a elemento de un array a otro).

```
int [] arrayFuente = {2, 3, 1, 5, 10};  
int [] arrayDestino = new int[arrayFuente.length];  
for (int i = 0; i < arrayFuente.length; i++) {  
    arrayDestino[i] = arrayFuente[i];  
}
```



Tema 4. Arrays

Arrays unidimensionales

□ Paso de arrays a métodos

- Cuando se pasa un array a un método, se pasa la **referencia** del array al método

```
1 package org.ip.tema04;
2 public class PasandoArraysAMetodos {
3     public static void main(String[] args) {
4         int x = 1;
5         int[] y = new int[10];
6         y[0] = 1;
7         System.out.println("x = " + x);
8         System.out.println("y[0] = " + y[0]);
9         metodo(x, y);
10        System.out.println("x = " + x);
11        System.out.println("y[0] = " + y[0]);
12    }
13    public static void metodo(int numero, int[] array) {
14        numero = 1001;
15        array[0] = 7777;
16    }
17 }
```

SALIDA

x = 1

y[0] = 1

x = 1

y[0] = 7777

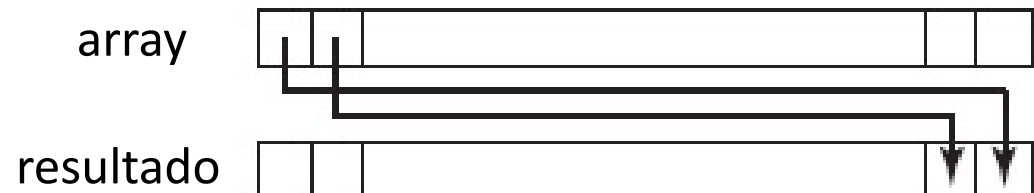
Tema 4. Arrays

➡ Arrays unidimensionales

❑ Devolver un array desde un método

- Cuando un método devuelve un array, se devuelve su **referencia**

```
public static int[] invertir(int[] array) {  
    int[] resultado = new int[array.length];  
    for (int i = 0, j = array.length - 1; i < array.length; i++, j--) {  
        resultado[j] = array[i];  
    }  
    return resultado;  
}
```



Tema 4. Arrays

Arrays unidimensionales

Error común

```
1 package org.ip.tema04;
2
3 public class Ejemplo1Arrays {
4     public static double media(int [] array) {
5         double suma = 0.0;
6         // Se dispara una excepcion
7         for (int i = 0; i <= array.length; i++) {
8             suma += (double)array[i];
9         }
10        return suma / array.length;
11    }
12
13    public static void main(String[] args) {
14        int [] temperaturas = {5, 10, 25, -4, 3};
15        double valorMedio = media(temperaturas);
16        System.out.println("La media es: " + valorMedio);
17    }
18 }
```

Accedemos a una posición que no existe

Se produce la excepción

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 5
    at org.ip.tema04.Ejemplo1Arrays.media(Ejemplo1Arrays.java:8)
    at org.ip.tema04.Ejemplo1Arrays.main(Ejemplo1Arrays.java:15)
```

Tema 4. Arrays

Arrays unidimensionales

Ejemplo. Calcular la media de los valores en un subarray temperaturas[inferior..superior]

```
public static double media(int [] temperaturas, int inferior, int superior) {  
    double suma = 0.0;  
    int elementos = superior - inferior + 1;  
    if (inferior < 0 || superior >= temperaturas.length || inferior > superior)  
        throw new RuntimeException ("Indices del subarray fuera de rango");  
    for (int i = inferior; i <= superior; i++){  
        suma += temperaturas[i] ;  
    }  
    return suma / elementos;  
}
```



```

1 package org.ip.tema04;
2 import java.util.Scanner;
3 public class Ejemplo2Arrays {
4     public static double media(int [] array, int inferior, int superior) {
5         if (inferior < 0 || superior >= array.length || inferior > superior)
6             throw new RuntimeException("Indices de array, fuera de rango");
7         double suma = 0.0;
8         int componentes = superior - inferior + 1;
9         for (int i = inferior; i <= superior; i++) {
10             suma += (double)array[i];
11         }
12         return suma / componentes;
13     }
14     public static void main(String[] args) {
15         int [] temperaturas = {5, 10, 25, -4, 3};
16         Scanner entrada = new Scanner(System.in);
17         int inferior = 0, superior = 0;
18         double valorMedio = 0.0;
19         boolean indicesCorrectos = false;
20         while (!indicesCorrectos) {
21             try
22                 System.out.println("Introduce indices inferior y superior"
23                     + "para calcular la media del subarray");
24                 inferior = entrada.nextInt();
25                 superior = entrada.nextInt();
26                 valorMedio = media(temperaturas, inferior, superior);
27                 indicesCorrectos = true;
28             } catch (RuntimeException e) {
29                 System.out.println(e.getMessage());
30             }
31         }
32         System.out.println("La media del subarray es: " + valorMedio);
33     }
34 }

```

Salida

```

Introduce indices inferior y superiorpara calcular la media del subarray
5 0
Indices de array, fuera de rango
Introduce indices inferior y superiorpara calcular la media del subarray
7 9
Indices de array, fuera de rango
Introduce indices inferior y superiorpara calcular la media del subarray
0 3
La media del subarray es: 9.0

```

Tema 4. Arrays

Arrays unidimensionales

☐ La clase Arrays

El paquete **java.util** contiene la clase **Arrays** que proporciona varios **métodos estáticos** (static) para realizar distintas operaciones con arrays, tales como:

- **Arrays.sort(a)** ordena los elementos del array a.
- **Arrays.equals(a, b)** comprueba si los arrays a y b son iguales.
- **Arrays.fill(a, valor)** rellena el array a con valor
- **Arrays.toString(a)** devuelve una cadena con el contenido del array.
- **Arrays.binarySearch(a, k)** busca el valor k en el array a. Es requisito que a esté ordenado.

Ejemplo de uso

```
1 package org.ip.tema04;
2
3 import java.util.Arrays;
4
5 public class Ejemplo3Arrays {
6
7     public static void main(String[] args) {
8         int [] array1 = {2, 4, 7, 10};
9         int [] array2 = {2, 4, 7, 10};
10        int [] array3 = {20, 30, 40, 50};
11        int valor = 50;
12        if (Arrays.equals(array1, array2))
13            System.out.println(Arrays.toString(array1) + " coincide con "
14                               + Arrays.toString(array2));
15        else
16            System.out.println(Arrays.toString(array1) + " no coincide con "
17                               + Arrays.toString(array2));
18        int posicion = Arrays.binarySearch(array3, valor);
19        if (posicion == 0)
20            System.out.println("El valor " + 50 + " no está en "
21                               + Arrays.toString(array3));
22        else
23            System.out.println("El valor " + valor + " está en la posición "
24                               + (posicion + 1) + " del array " + Arrays.toString(array3));
25    }
26 }
```

Salida

[2, 4, 7, 10] coincide con [2, 4, 7, 10]

El valor 50 está en la posición 4 del array [20, 30, 40, 50]

Tema 4. Arrays

Arrays multidimensionales

- ☐ Introducción
- ☐ Declaración y creación de un array multidimensional
- ☐ Acceso a los elementos de un array multidimensional
- ☐ Procesamiento en un array multidimensional

Tema 4. Arrays

Arrays bidimensionales

Introducción

En el apartado anterior hemos estudiado cómo usar un **array unidimensional** para almacenar una **colección lineal** de elementos. Podemos usar un **array bidimensional** para guardar una **matriz** o una tabla. Por ejemplo, la tabla siguiente describe las distancias entre ciudades, podemos usar un array bidimensional para almacenarlas.

	Chicago	Boston	New York	Atlanta	Miami	Dallas	Houston
Chicago	0	983	787	714	1375	967	1087
Boston	983	0	214	1102	1763	1723	1842
New York	787	214	0	888	1549	1548	1627
Atlanta	714	1102	888	0	661	781	810
Miami	1375	1763	1549	661	0	1426	1187
Dallas	967	1723	1548	781	1426	0	239
Houston	1087	1842	1627	810	1187	239	0

Tema 4. Arrays

Arrays bidimensionales

☐ Declaración y creación

```
tipo [][] identificador = new tipo[elementos1][elementos2];
```

Ejemplos

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

```
int [][] matrizA = {{4,5,9},{10,14,25}, {7,8,15}}
```

Equivalente a:

```
int [][] matrizA = new int[3][3];
```

```
matrizA[0][0]= 4; matrizA[0][1]= 5; matrizA[0][2]= 9;
```

```
matrizA[1][0]= 10; matrizA[1][1]= 14; matrizA[1][2]= 25;
```

```
matrizA[2][0]= 7; matrizA[2][1]= 8; matrizA[2][2]= 15;
```

matrizA



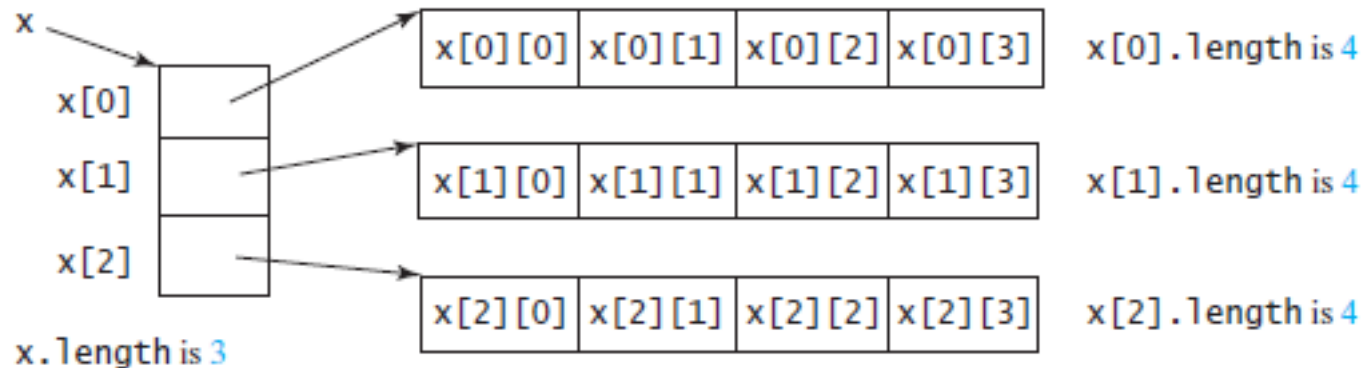
4	5	9
10	14	25
7	8	15

Tema 4. Arrays

➡ Arrays bidimensionales

Un array bidimensional lo podemos ver como un array unidimensional en el cual cada elemento es otro array unidimensional.

Por ejemplo: `int [][] x = new int[3][4];` podemos representarlo como sigue,

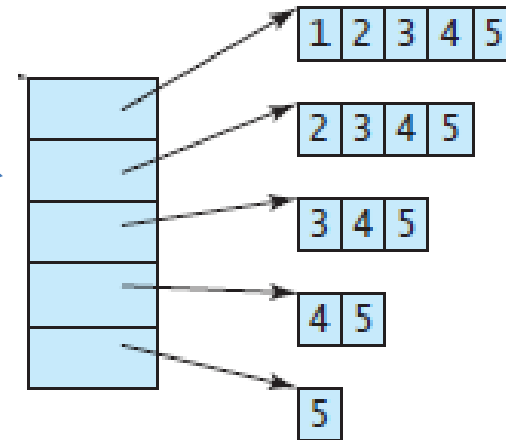


Tema 4. Arrays

➡ Arrays bidimensionales

Cada fila en el array bidimensional es a su vez un array unidimensional, por tanto, las filas pueden tener diferentes longitudes. Un array de este tipo se conoce como *array desigual*. Por ejemplo:

```
int [][] trianguloArray = {  
    {1,2,3,4,5},  
    {2,3,4,5},  
    {3,4,5},  
    {4,5},  
    {5}  
};
```



Equivalente a: `int [][] trianguloArray = new int[5][]; trianguloArray[0] = new int[5];
trianguloArray[1] = new int[4]; trianguloArray[2] = new int[3]; etc.`

Tema 4. Arrays

➡ Arrays bidimensionales

□ Acceso a los elementos de un array bidimensional

Para acceder a los elementos de un array utilizamos índices, en este caso necesitaremos dos que indican la *fila* y la *columna*.

```
int [][] matriz = new int[5][5];  
matriz[2][1] = 7;
```

	[0]	[1]	[2]	[3]	[4]
[0]					
[1]					
[2]		7			
[3]					
[4]					

Tema 4. Arrays

Arrays bidimensionales

Procesamiento en un array bidimensional

Si se van a tratar todos los elementos del array o parte de ellos que ocupan posiciones consecutivas utilizaremos **dos bucles for**, uno para indicar las *filas* y el otro para las *columnas*. Vamos a ver algunos ejemplos:

Ejemplo 1. Inicializar el array con valores que introducimos por teclado.

```
int [][] matriz = new int [3][3];
```

```
Scanner entrada = new Scanner(System.in);
```

```
System.out.println("Vamos a crear una matriz de " + matriz.length + " filas y " +  
matriz[0].length + " columnas");
```

```
for (int fila = 0; fila < matriz.length; fila++) {  
    for (int columna = 0; columna < matriz[fila].length ; columna++) {  
        matriz[fila][columna] = entrada.nextInt();}}
```

Tema 4. Arrays

Haciendo uso de un método:

```
public static int [][] leerEnteros2D() {    // Lectura matriz cuadrada
    Scanner entrada = new Scanner(System.in);
    System.out.println("Introduce el número de filas y columnas de la matriz");
    int dimension = entrada.nextInt();
    int [][]a = new int [dimension][dimension];
    System.out.println("Introduce valores enteros en la matriz ");
    for (int i = 0; i < a.length; i++) {
        for (int j = 0; j < a[i].length; j++) {
            System.out.print("Introduce valor a[" + i + ", " + j + "] => ");
            a[i][j] = entrada.nextInt();
        }
    }
    return a;
}
```

Tema 4. Arrays

Haciendo uso de un método:

```
public static int [][] leerEnteros2D() {    // Lectura matriz no cuadrada
    Scanner entrada = new Scanner(System.in);
    System.out.println("Introduce el número de filas");
    int dimension1 = entrada.nextInt();
    System.out.println("Introduce el número de columnas");
    int dimension2 = entrada.nextInt();
    int [][]a = new int [dimension1][dimension2];
    System.out.println("Introduce valores enteros en la matriz ");
    for (int i = 0; i < a.length; i++) {
        for (int j = 0; j < a[i].length; j++) {
            System.out.print("Introduce valor a[" + i + ", " + j + "] => ");
            a[i][j] = entrada.nextInt();
        }
    }
    return a;}
```

Tema 4. Arrays

Arrays bidimensionales

☐ Procesamiento en un array bidimensional

Ejemplo 2. Inicializar el array con valores aleatorios entre 0 y 99.

```
int [][] matriz = new int [3][3];  
for (int fila = 0; fila < matriz.length; fila++){  
    for (int columna = 0; columna < matriz[fila].length ; columna++) {  
        matriz[fila][columna] = (int) (Math.random() * 100);  
    }  
}
```

Tema 4. Arrays

Arrays bidimensionales

☐ Procesamiento en un array bidimensional

Ejemplo 3. Mostrar el array por pantalla.

```
System.out.println("La matriz creada es ");  
for (int fila = 0; fila < matriz.length; fila++) {  
    for (int columna = 0; columna < matriz[fila].length; columna++) {  
        System.out.print(matriz[fila][columna] + "\\t");  
    }  
    System.out.println();  
}
```

Tema 4. Arrays

Arrays bidimensionales

☐ Procesamiento en un array bidimensional

Ejemplo 4. Sumar todos los elementos del array.

```
int sumaTotal = 0;
for (int fila = 0; fila < matriz.length; fila++) {
    for (int columna = 0; columna < matriz[fila].length; columna++) {
        sumaTotal += matriz[fila][columna];
    }
}
```

Tema 4. Arrays

Arrays bidimensionales

☐ Procesamiento en un array bidimensional

Ejemplo 4. Sumar los elementos del array por columnas.

```
for (int columna = 0; columna < matriz[0].length; columna++) {  
    int sumaColumna = 0;  
    for (int fila = 0; fila < matriz.length; fila++) {  
        sumaColumna += matriz[fila][columna];  
    }  
    System.out.println("Suma columna " + columna + " = " + sumaColumna);  
}
```


Tema 4. Arrays

Arrays bidimensionales

Ejemplo 5. Obtener qué fila tiene la suma más grande.

```
int maxSumaFila = 0;
int indiceMaxSumaFila = 0;
for (int columna = 0; columna < matriz[0].length; columna++) {
    maxSumaFila += matriz[0][columna];
}
for (int fila = 1; fila < matriz.length; fila++) {
    int sumaFilaActual = 0;
    for (int columna = 0; columna < matriz[fila].length; columna++) {
        sumaFilaActual += matriz[fila][columna];
    }
    if (sumaFilaActual > maxSumaFila) {
        maxSumaFila = sumaFilaActual;    indiceMaxSumaFila = fila;
    }
}
System.out.println("Fila " + indiceMaxSumaFila + " tiene la maxima suma de " + maxSumaFila);
```

Tema 4. Arrays

Arrays bidimensionales

Ejemplo 6. Aleatorizar el contenido de una matriz (random suffling).

```
for (int i = 0; i < matriz.length; i++) {  
    for (int j = 0; j < matriz[i].length; j++)  
        int i1 = (int)(Math.random() * matriz.length);  
        int j1 = (int)(Math.random() * matriz[i].length);  
        double temp = matriz[i][j] ;  
        matriz[i][j] = matriz[i1][j1];  
        matriz[i1][j1] = temp;  
    }  
}
```

Tema 4. Arrays

Arrays bidimensionales

- ❑ **Paso de matrices a métodos** \Rightarrow Cuando se pasa una matriz a un método, se pasa la referencia de la matriz al método
- ❑ **Devolver un array desde un método** \Rightarrow Cuando un método devuelve una matriz, se devuelve su referencia

```
23 public static int suma(int[][] matriz) {  
24     int sumaTotal = 0;  
25     for (int fila = 0; fila < matriz.length; fila++) {  
26         for (int columna = 0; columna < matriz[fila].length; columna++) {  
27             sumaTotal += matriz[fila][columna];  
28         }  
29     }  
30     return sumaTotal;  
31 }  
32 }
```

Tema 4. Arrays

```
1 package org.ip.tema04;
2
3 import java.util.Scanner;
4
5 public class PasandoDevolviendoMatricesMetodos {
6
7     public static void main(String[] args) {
8         int[][] matriz = getMatriz();
9         System.out.println("\nLa suma de todos los elementos es " + suma(matriz));
10    }
11
12    public static int[][] getMatriz() {
13        Scanner entrada = new Scanner(System.in);
14        int[][] matriz = new int[3][4];
15        System.out.println("Introduce " + matriz.length + " filas and "
16            + matriz[0].length + " columns: ");
17        for (int i = 0; i < matriz.length; i++)
18            for (int j = 0; j < matriz[i].length; j++)
19                matriz[i][j] = entrada.nextInt();
20        return matriz;
21    }
```

Tema 4. Arrays

Ejemplo 7. Ejemplo de un método que muestra los valores de una matriz 2D

```
public static void mostrar2D(int [][] a) {  
    System.out.println("Los valores guardados en la matriz son ");  
    for (int i = 0; i < a.length; i++) {  
        for (int j = 0; j < a[i].length; j++) {  
            System.out.print(a[i][j] + "\\t");  
        }  
        System.out.println();  
    }  
}
```

Tema 4. Arrays

Ejemplo 8. Método de clase para comprobar si una matriz es simétrica.

```
public static boolean esSimetrica(int [][] matriz) {  
    boolean simetria = true;  
    int fila = 0; int columna;  
    while (simetria && (fila < matriz.length)) {  
        columna = 0;  
        while (simetria && (columna < matriz[fila].length)) {  
            if (matriz[fila][columna] != matriz[columna][fila])  
                simetria = false;  
            columna++;  
        }  
        fila++;  
    }  
    return simetria;  
}
```

Salida

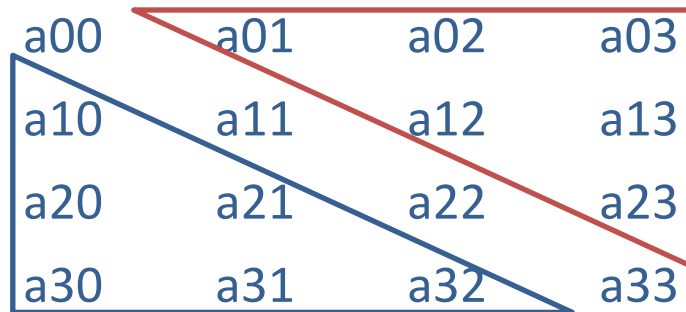
La matriz es simétrica

```
1 package org.ip.tema04;
2
3 public class Ejemplo4Matriz {
4
5     public static boolean esSimetrica(int [][] matriz) {
6         boolean simetrica = true;
7         int fila = 0;
8         int columna;
9         while (simetrica && (fila < matriz.length)) {
10             columna = 0;
11             while (simetrica && (columna < matriz[fila].length)) {
12                 if (matriz[fila][columna] != matriz[columna][fila])
13                     simetrica = false;
14                 columna++;
15             }
16             fila++;
17         }
18         return simetrica;
19     }
20
21     public static void main(String[] args) {
22         int [][] matriz = {{1, 2, 3}, {2, 1, 4}, {3, 4, 1}};
23         boolean simetrica = esSimetrica(matriz);
24         if (simetrica)
25             System.out.println("La matriz es simétrica");
26         else
27             System.out.println("La matriz no es simétrica");
28     }
29 }
```

Tema 4. Arrays

Ejemplo 9. Otro método de clase para comprobar si una matriz es simétrica.

Otra solución sería comparar solo los elementos de la triangular superior con los de la triangular inferior o al contrario. Con ello, disminuiríamos el número de iteraciones de los bucles. Consideremos por ejemplo los elementos de la triangular superior, bastaría comparar los incluidos en el triángulo rojo con los del triángulo azul.



a00	a01	a02	a03
a10	a11	a12	a13
a20	a21	a22	a23
a30	a31	a32	a33

En este caso se cumple:

El índice para la fila empieza en 0 y termina en el número de filas menos 1.

El índice para la columna empieza en la fila +1 y termina en el número de columnas.

Tema 4. Arrays

```
public static boolean esSimetrica(int [][] matriz) {  
    boolean simetria = true;  
    int fila = 0; int columna;  
    while (simetria && (fila < matriz.length - 1)) {  
        columna = fila + 1;  
        while (simetria && (columna < matriz[fila].length)) {  
            if (matriz[fila][columna] != matriz[columna][fila])  
                simetria = false;  
            columna++;  
        }  
        fila++;  
    }  
    return simetria;  
}
```

Tema 4. Arrays

Ejemplo 10. Método de clase para obtener el producto de dos matrices cuadradas.

Para calcular el producto de las matrices **A** (de dimensiones $m \times p$) y **B** (de dimensiones $p \times n$):

$$A_{m \times p} \times B_{p \times n} = C_{m \times n}$$

Utilizamos la expresión:

$$c_{ij} = \sum_{k=1}^p a_{ik} * b_{kj}$$

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{pmatrix} \times \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{pmatrix} = \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix}$$

$$c_{11} = a_{11}b_{11} + a_{12}b_{21} + a_{13}b_{31}$$

$$c_{12} = a_{11}b_{12} + a_{12}b_{22} + a_{13}b_{32}$$

.....

En nuestro caso, las matrices son cuadradas $\Rightarrow m = p = n$

Tema 4. Arrays

```
public static int [][] producto(int [][] a, int [][] b) { // OJO! Matrices cuadradas
    if (a.length != b.length) throw new RuntimeException("Las dimensiones no
        coinciden, no se puede realizar la operación");
    int [][] c = new int [a.length][b.length];
    for (int i = 0; i < a.length; i++) {
        for (int j = 0; j < b.length; j++) {
            c[i][j] = 0;
            for (int k = 0; k < b.length; k++) { // Desarrolla la sumatoria
                c[i][j] += a[i][k] * b[k][j];
            }
        }
    }
    return c;
}
```

¡MUCHAS GRACIAS!



UNIVERSIDAD DE ALMERÍA

