

TEORÍA

Muestra el contenido del Array.

- **Arrays.toString(nombreVariable);**

NEW. Añadir una variable

NULL. Anula los valores (“vacío”).

BÚSQUEDA EN TABLA NO ORDENADA (P.139) →

FILL: Pone el mismo valor en todo el Array

- **Arrays.fill();**

```
/* búsqueda secuencial */
int indiceBusqueda = 0; //índice que usamos para recorrer la tabla
while (indiceBusqueda < t.length && //no es el último elemento
      t[indiceBusqueda] != claveBusqueda) { //y no encontrado
    indiceBusqueda++; //incrementamos el índice de búsqueda
}
if (indiceBusqueda < t.length) {
    ... //claveBusqueda se encuentra en la posición indiceBusqueda
} else { //el índice se ha salido de rango
    ... //no encontrado
}
```

SORT (FUNCIONES INTERESANTES): Ordena en Array (P.138)

- **Arrays.sort();**

binarySearch (BÚSQUEDA EN TABLA ORDENADA)

Te dice dónde lo encuentra y si está o no en el Array, si da negativo es que no está en el Array. En caso de que apareciera, te coloca la posición en la que aparecería, para que estuvieran los valores ordenados.

COPIOF (INTERESANTE) P.142 =

- **copyOf (origen[], int longitud)**

Devuelve la primera ocurrencia.

Copiar Arrays (para hacer trampa)

Veamos un ejemplo:

```
int t[] = {1, 2, 1, 6, 23}; //tabla origen
int a[], b[]; // tablas destino
a = Arrays.copyOf(t, 3); //a = [1, 2, 1]
b = Arrays.copyOf(t, 10); //b = [1, 2, 1, 6, 23, 0, 0, 0, 0, 0]
```

FOR EACH

Recorrido estructuras, no se le da ni el inicio ni el fin, tiene dos partes en el for, a la derecha es el array que recorres (derecha de los puntos), y a la izquierda el mismo tipo de array (Ej. Int)

```
14
15 //Pintamos de izquierda a dcha: 4 5 -2 9
16 //Opcion 1: Para perezosos
17 System.out.println(Arrays.toString(prueba));
18
19 //Opción 2: For clásico
20 for (int a=0;a<prueba.length;a++)
21     System.out.print(prueba[a]+ " ");
22
23 System.out.println("");
24
25 //Opción 3: For each ( for extendido)
26 for (int a:prueba)
27     System.out.print(a+ " ");
28 System.out.println("");
29
30 //Pintamos de dcha a izqda: 9 -2 5 4
```

```
Actividad_5.java Pasoporvalor.java EMAILVÁLIDO.java Ejercicio
1 package Tema5;
2
3 import java.util.Arrays;
4
5 public class ComparaArrays {
6
7     public static void main(String[] args) {
8
9         int a[] = {3,7};
10        int b[] = {3,7};
11
12
13        if (Arrays.equals(a, b)) {
14            System.out.println("Son iguales");
15        }
16        else System.out.println("No son iguales");
17
18    }
19
20 }
21
```

18/11/2022 COMPARACIÓN DE DOS TABLAS

Equals = comparar String, devuelve un boolean, cuando ambos arrays son iguales, comparamos con dos arrays enteros (t1, t2)

Static boolean equals (tipo a[], tipo b[])

TABLAS DOS DIMENSIONES P.150

Pones dos corchetes [], []. Primero filas y luego columnas.

Columnas

Filas

Ej.

Datos [2] [1] = 7;

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

y con ello, estamos reservando espacio en la memoria para (5 × 5) 25 elementos.

	0	1	2	3	4
0	2	4	6	12	0
1	2	-11	4	7	86
2	0	1	6	5	3
3	1	93	6	-2	0
4	9	71	23	2	8

↑ eje y
← eje x

Figura 5.19. Matriz de 5 × 5 elementos. Ahora la identificación de cada elemento viene dada por el índice del eje X y el índice del eje Y.

Y = columnas

X = filas

Utiliza bucles
anidados

Los algoritmos que utilizan matrices requieren dos bucles anidados. Un bucle se encarga del índice para la dimensión X y el otro para el índice del eje Y. Veamos un ejemplo para introducir por teclado la matriz `datos`:

```
for (i = 0; i < 5; i++) { //eje X
    for (j = 0; j < 5; j++) { //eje Y
        datos[i][j] = sc.nextInt(); //leemos el elemento [i][j]
    }
}
```

Y = 0, se queda estático

2 -1 7 8 20

i=0

1 1 1 1 1

datos[0][j]

datos[0][0] = 2
datos[0][1] = -1
datos[0][2] = 7
datos[0][3] = 8
datos[0][4] = 20

i

i=1

datos[1][j]

datos[1][0] = 1
datos[1][1] = 1
datos[1][2] = 1
datos[1][3] = 1
datos[1][4] = 1

COMO SE PINTA UN ARRAYS

Dos for con la "i" y "j", y añades un "if", y haces un Syso, con los valores que se coge, mientras una de las variables se queda estática.

SEGUNDA FORMA DE HACERSE

- **System.out.println (Arrays.deepToString (datos));**

CON UN FOR EACH

```
System.out.println(Arrays.deepToString(t)); //mostramos
//otra forma de mostrar es hacerlo recorriendo nosotros la matriz.
//Una matriz es un conjunto de filas (tabla unidimensional). Y cada fila
//está compuesta por una serie de elementos.
for (int fila[] : t) {
    for (int columna: fila) {
        System.out.print(columna + " ");
    }
    System.out.println();
}
```