# **TEORÍA**

### Muestra el contenido del Array.

### Arrays.toString(nombreVariable);

```
NEW. Añadir una variable
                                              /* búsqueda secuencial */
                                              int indiceBusqueda = 0; //indice que usamos para recorrer la tabla
NULL. Anula los valores ("vacío").
                                              while (indiceBusqueda < t.length && //no es el último elemento
                                                 t[indiceBusqueda] != claveBusqueda) { //y no encontrado
BÚSOUEDA
                                                 indiceBusqueda++; //incrementamos el índice de búsqueda
                EN
                        TABLA
                                    NO
ORDENADA (P.139)
                                              if (indiceBusqueda < t.length) {
                                                 ... //claveBusqueda se encuentra en la posición indiceBusqueda
FILL: Pone el mismo valor en todo el
                                              } else { //el índice se ha salido de rango
Array
                                                 ... //no encontrado
```

Arrays.fill();

**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 =**

• copiOf (origen[], int longuitud)

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
   //Opcion 1: Para perezosos
16
   System.out.println(Arrays.toString(prueba));
17
18
19
    //Opción 2: For clásico
   for (int a=0;a<prueba.length;a++)</pre>
20
        System.out.print(prueba[a]+ " ");
21
22
23
   System.out.println("");
24
25
    //Opción 3: For each (for extendido)
26
   for (int a:prueba)
        System.out.print(a+ " ");
27
28
   System.out.println("");
29
    //Dintamos de deha a izada: Q _7 5 /
```

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

# DE Ejercicia 18/11/2022 COMPARACIÓN DE DOS TABLAS

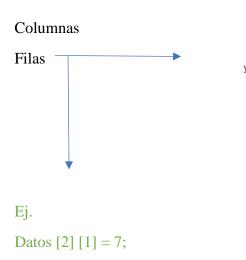
**Equals** = comparar String, devuelve un boolean, cuando ambos arrays son iguales, comparamos con dos arrays enteros (t1, t2)

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

## TABLAS DOS DIMENSIONES P.150

int datos[][]:

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



```
\mathtt{datos} = \mathtt{new} \ \mathtt{int}[5][5]; y con ello, estamos reservando espacio en la memoria para (5 \times 5) 25 elementos.
```

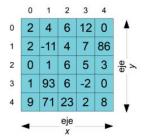


Figura 5.19. Matriz de 5 x 5 elementos. Ahora la identificación de cada elemento viene dada por el índice del eje X y el índice del eje Y.

```
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]
  }
}</pre>
```

```
Y = 0, se queda estático
               i=0
2 -1 7 8 20
                                                 i=1
11111
              datos[0][j]
                                                 datos[1][j]
               datos[0][0] = 2
                                                 datos[1][0] = 1
               datos[0][1] = -1
                                                 datos[1][1] = 1
                                                 datos[1][2] = 1
               datos[0][2] = 7
                                                 datos[1][3]=1
               datos[0][3] = 8
                                                 datos[1][4] = 1
               datos[0][4] = 20
```

### **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();
}
```