Definición: tipo [] nombre;

**Inicalización**: nombre= new tipo[longitud]; nombre={elemento1, elemento2,... elementoN};

Índices: de 0 a N-1

Acceso o modificación de un dato: nombre\_array[índice]=valor;

**Recorrido**: for(int intIndice=0; intIndice<nombre\_array.length; intIndice++) for(tipo nombre\_variable:nombre\_array)

System.out.println(nombre\_array [intIndice]); System.out.println(nombre\_variable);

Paso como parámetros a funciones: se pasa la ubicación de la variable, por tanto cualquier cambio en la función, cambiará el parámetro real

fill (nombreArray, desde, hasta, valor): rellena un array con el valor indicado. Lo indicado en cursiva no es obligatorio, pero puede limitar desde y hasta donde queremos rellenar de un array.

Arrays.toString(nombre\_array): Convierte un array en una cadena de texto que se puede imprimir

Arrays.sort(nombre\_array): Ordena un array. Esta operación cuesta mucho trabajo

## Búsqueda array desordenado:

**Búsqueda array ordenado binarySearch (nombre\_array[], desde, hasta, claveBusqueda):** devuelve el índice donde se encuentra el elemento. En caso de no encontrarlo, devuelve en negativo el índice del siguiente elemento donde se debería insertar el elemento. Si hay varias ocurrencias dentro de un array, no se garantiza que se apunte a la primera. Desde y hasta vuelven a no ser obligatorios

Arrays.copyOf (nombre\_array, longitud): Devuelve una copia del array de la longitud indicada

Arrays.copyOfRange (nombre\_array, desde, hasta): devuelve un array que tiene los elementos desde el índice indicado hasta-1

System.arrayCopy (array\_origen, índice\_origen, array\_destino, índice\_destino, número\_de\_elementos\_a\_copiar): copia en el índice índice\_destino, dentro de array\_destino, los número\_de\_elementos\_a\_copiar, desde array\_origen y la posición índice\_origen

```
Inserción en array no ordenado: Inserta en la última posición del array el nuevo elemento
nombre_array = Arrays.copyOf(nombre_array, (nombre_array.length+1));
nombre_array[nombre_array.length-1]=valor_Insertado;
```

```
Inserción en array ordenado: buscamos el índice de inserción del nuevo elemento. Redimensionamos el array con una posición más. Movemos los elementos desde la posición de inserción, una posición a la derecha. Insertamos el elemento en la posición de inserción.
```

```
intIndiceInsercion=Arrays.binarySearch(nombreArray, valorInsertar);
if(intIndiceInsercion<0)</pre>
      intIndiceInsercion=-intIndiceInsercion-1;
nombreArray =Arrays.copyOf(nombreArray, nombreArray.length+1);
System.arraycopy(nombreArray, intIndiceInsercion, nombreArray, intIndiceInsercion+1,
nombrArray.length-(intIndiceInsercion+1));
nombreArray [intIndiceInsercion]=intValor;
Borrado array no ordenado: buco el elemento y si existe cambio el último por este elemento. Luego redimensiono el
array a uno menos
intIndice = buscar(nombreArray, valorBuscado);
if(intIndice !=-1){
      nombreArray [intIndice] = nombreArray [nombreArray.length-1];
      nombreArray =Arrays.copyOf(nombreArray, nombreArray.length-1);
}
Borrado array ordenado: buco el elemento y si existe, muevo los elementos a la derecha de este elemento, una
posición a la izquierda. Luego redimensiono el array a uno menos
intIndiceBorrado = Arrays.binarySearch(nombreArray, valorElementoABorrar);
if (intIndiceBorrado >= 0) {
      System.arraycopy(nombreArray, intIndiceBorrado + 1, nombreArray, intIndiceBorrado,
     nombreArray.length - (intIndiceBorrado + 1));
      nombreArray = Arrays.copyOf (nombreArray, nombreArray.length - 1);
```

## **Arrays multidimensionales**

**Definición**: se colocan tantos corchetes como dimensiones tenga el array. tipo[][][] nombre\_array;

Inicalización: nombre\_array= new tipo[longitud1]...[longitudN]; nombre={...{elemento1, elemento2,... elementoN}}; No es conveniente inicializar con más de dos dimensiones mediante llaves

Índices: de 0 a N-1

}

Acceso o modificación de un dato: nombre\_array[índice1]... [índiceN]=valor;

```
Recorrido: hay que meter un for o for-each por cada dimensión for(int intlndice1=0; intlndice1<nombre_array.length; intlndice1++) for(tipo[]...[] nombre_array2:nombre_array) ...

for(int intlndiceN=0; intlndiceN<nombre_array[0]...[0].length; intlndice1++) for(tipo nombre_variable:nombre_arrayN)

System.out.println(nombre_array[intlndice1] [intlndiceN]); System.out.println(nombre_variable);
```