

CURSO  
BACKEND 1

Arreglos y vectores

# Material de lectura

## OBJETIVOS DE LA GUÍA

En esta guía aprenderemos a:

- Definir de vectores y matrices
- Llenar de vectores y matrices
- Mostrar vectores y matrices



egg



Argentina  
programa  
4.0

## Arreglos: vectores y matrices

Un arreglo es un contenedor de objetos que tiene un número fijo de valores del mismo tipo. El tamaño del arreglo es establecido cuando el arreglo es creado y luego de su creación su tamaño es fijo, esto significa que no puede cambiar. Cada una de los espacios de un arreglo es llamada elemento y cada elemento puede ser accedido por un índice numérico que arranca desde 0 hasta el tamaño menos uno. Por ejemplo, si tenemos un vector de 5 elementos los índices serían: 0-1-2-3-4.

Al igual que la declaración de otros tipos de variables, la declaración de un arreglo tiene dos componentes: el tipo de dato del arreglo y su nombre. El tipo de dato del arreglo se escribe como `tipo[]`, en donde, `tipo` es el tipo de dato de cada elemento contenido en él. Los corchetes sirven para indicar que esa variable va a ser un arreglo. El tamaño del arreglo no es parte de su tipo (es por esto que los corchetes están vacíos).

Una vez declarado un arreglo hay que crearlo/dimensionarlo, es decir, hay que asignar al arreglo un tamaño para almacenar los valores. La creación de un arreglo se hace con el operador `new`. Recordemos que las matrices son bidimensionales por lo que tienen dos tamaños, uno para las filas y otro para las columnas de la matriz.

### ***Declaración y creación de un Vector***

```
tipo[] arregloV = new tipo[Tamaño];
```

### ***Declaración y creación de una Matriz***

```
tipo[][] arregloM = new tipo[Filas][Columnas];
```



¿NECESITAS UN EJEMPLO?

```
public static void main(String[] args) {  
  
    // Creo un arreglo llamado vector con dimensión 5 que  
    // solo pueda alojar números enteros  
    int[] vector = new int[5];  
  
    // Creo una matriz con dimensión 3x3 que  
    // solo pueda alojar cadenas  
    String[][] matriz = new String[3][3];  
  
}
```



¡MANOS A LA OBRA!

## Ejercicio 13

Crea un vector llamado 'Equipo' cuya dimensión sea la cantidad de compañeros de equipo y define su tipo de dato de tal manera que te permita alojar sus nombres más adelante.



Revisemos lo aprendido hasta aquí

- Crear vectores y matrices que puedan alojar distintos tipos de dato

## Asignar elementos a un arreglo

Cuando queremos ingresar un elemento en nuestro arreglo vamos a tener que elegir el subíndice en el que lo queremos guardar. Una vez que tenemos el subíndice decidido tenemos que invocar nuestro vector por su nombre y entre corchetes el subíndice en el que lo queremos guardar.

Después, pondremos el signo de igual (que es el operador de asignación) seguido del elemento a guardar. En las matrices vamos a necesitar dos subíndices y dos corchetes para representar la posición de la fila y la columna donde queremos guardar el elemento.

### Asignación de un Vector

```
vector[0] = 5;
```

### Asignación de una Matriz

```
matriz[0][0] = 6;
```

Esta forma de asignación implica asignar todos los valores de nuestro arreglo de uno en uno, esto va a conllevar un trabajo bastante grande dependiendo del tamaño de nuestro arreglo.

Entonces, para poder asignar varios valores a nuestro arreglo y no hacerlo de uno en uno usamos un bucle **Para**. El bucle Para, al poder asignarle un valor inicial y un valor final a una variable, podemos adaptarlo fácilmente a nuestros arreglos. Ya que, pondremos el valor inicial de nuestro arreglo y su valor final en las respectivas partes del Para. Nosotros usamos la variable creada en el Para, y la pasaremos a nuestro arreglo para representar todos los subíndices del arreglo, de esa manera, recorriendo todas las posiciones y asignándole a cada posición un elemento.

Para poder asignar varios elementos a nuestra matriz, usamos dos bucles **Para** anidados., ya que un **Para** recorrerá las filas (*variable i*) y otro las columnas (*variable j*).

### Asignación de un Vector

```
for (int i = 0; i < 5; i++) {  
    vector[i] = 5;  
}
```

### Asignación de una Matriz

```
for (int i = 0; i < 3; i++) {  
    for (int j = 0; j < 3; j++) {  
        matriz[i][j] = 6;  
    }  
}
```



Pueden encontrar un ejemplo de vectores y matrices en tu Aula Virtual.

### Vector:

```
public static void main(String[] args) {  
  
    int vector[] = new int[5]; // Le ponemos la dimension al  
    vector  
  
    // Puedo asignar valores de esta manera  
    vector[0] = 3;  
  
    // Asigno valores mediante el for  
    for (int i = 0; i < 5; i++) {  
  
        vector[i] = i + 3;  
  
    }  
  
    // Muestro el vector  
    for (int i = 0; i < 5; i++) {  
        System.out.println "[" + vector[i] + "]" ;  
    }  
  
}
```

## Matriz:

```
public static void main(String[] args) {

    String[] [] matriz = new String [3] [3];

    // Puedo asignar valores de esta manera
    matriz[0][0] = "A";

    // Asigno valores mediante el For
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            matriz[i][j] = "A";
        }
    }

    // Muestro la matriz
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {

            System.out.print "[" + matriz[i][j] + " ");

        }
        System.out.println("");
    }

}
```



**¡MANOS A LA OBRA!**

## Ejercicio 14

Utilizando un Bucle for, aloja en el vector Equipo, los nombres de tus compañeros de equipo



### Revisemos lo aprendido hasta aquí

- Asignar valores a vectores y matrices de forma manual o con bucles.

## Vectores y matrices en subprogramas

Los arreglos se pueden pasar como parámetros a un subprograma (función o procedimiento) del mismo modo que pasamos variables, poniendo el tipo de dato delante del nombre del vector o matriz, pero deberemos sumarle las llaves para representar que es un vector o matriz. Sin embargo, hay que tener en cuenta que la diferencia entre los arreglos y las variables, es que los arreglos siempre se pasan por referencia.

```
public static void llenarVector(int[] vector){  
  
}  
  
public static void mostrarMatriz(int[][] matriz){  
  
}
```

A diferencia de Pseint, en Java si podemos devolver un vector o una matriz en una función para usarla en otro momento. Lo que hacemos es poner como tipo de dato de la función, el tipo de dato que tendrá el vector y así poder devolverlo.

```
public static int devolverVector(){  
  
    int[] vector = new int[5];  
  
    return vector;  
}
```



#### Revisemos lo aprendido hasta aquí

- Operar con vectores y matrices desde funciones o procedimientos.