PROMETEO

PROGRAMACIÓN

Arrays

Arrays

Un **array** en Java es una estructura de almacenamiento estática que almacena un conjunto de elementos del **mismo tipo de dato** en posiciones contiguas de memoria.

Los arrays se dividen en:

1. **Arrays unidimensionales**: Almacenan elementos en una sola fila.
2. **Arrays multidimensionales**: Almacenan elementos en dos o más dimensiones, como matrices o tablas.

01

Declaración

Declaración de Arrays Unidimensionales

La declaración de un array unidimensional especifica el tipo de datos y el nombre del array.

Sintaxis:

```
tipo[] nombreArray;   // Recomendado
tipo nombreArray[];   // Alternativa
```

Ejemplo: Declaración de Arrays

```
int[] numeros;      // Declaración de un array de enteros
double[] precios;   // Declaración de un array de números decimales
String[] nombres;   // Declaración de un array de cadenas
```

01

Declaración

Declaración de Arrays Multidimensionales

Los arrays multidimensionales se declaran con varios pares de corchetes ([]).

Sintaxis:

```
tipo[][] nombreArray;   // Array bidimensional (matriz)
tipo[][][] nombreArray; // Array tridimensional
```

Ejemplo: Declaración de Arrays Multidimensionales

```
java
Copiar código
int[][] matriz;       // Declaración de un array bidimensional
String[][] tabla;     // Declaración de una tabla de cadenas
double[][][] cubo;    // Declaración de un array tridimensional
```

02

Creación

Una vez declarado un array, debe ser **creado** asignándole un tamaño fijo.

Creación de Arrays Unidimensionales

- La creación de un array asigna memoria para los elementos del tipo especificado.

Sintaxis:

```
nombreArray = new tipo[tamaño];
```

Ejemplo: Creación de Arrays

```
int[] numeros = new int[5];  // Array con espacio para 5 enteros
String[] nombres = new String[3];  // Array para 3 cadenas
```

02

Creación

Creación de Arrays Multidimensionales

- Para crear arrays multidimensionales, se deben definir el número de filas y columnas.

Sintaxis:

```
nombreArray = new tipo[filas][columnas];
```

Ejemplo: Creación de Arrays Multidimensionales

```
int[][] matriz = new int[3][3];   // Matriz de 3x3
String[][] tabla = new String[2][4];  // Tabla de 2 filas y 4 columnas
```

03

Inicialización

La inicialización asigna **valores** a los elementos del array.

1. **Inicialización de Arrays Unidimensionales**
   - ➢ Inicialización Directa:

```
java
Copiar código
int[] numeros = {1, 2, 3, 4, 5};
```

   - ➢ Asignación Individual:

```
int[] numeros = new int[3];
numeros[0] = 10;
numeros[1] = 20;
numeros[2] = 30;
```

03

Inicialización

La inicialización asigna **valores** a los elementos del array.

1. **Inicialización de Arrays Multidimensionales**
   - ➢ Inicialización Directa:

```
int[][] matriz = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
```

   - ➢ Asignación Individual:

```
int[][] matriz = new int[2][2];
matriz[0][0] = 1;
matriz[0][1] = 2;
matriz[1][0] = 3;
matriz[1][1] = 4;
```

04

Acceso

La inicialización asigna **valores** a los elementos del array.

1. **Acceso en Arrays Unidimensionales**

```java
int[] numeros = {10, 20, 30};
System.out.println("Primer elemento: " + numeros[0]);  // Accede al índice 0
```

2. **Acceso en Arrays Multidimensionales**

```
java
Copiar código
int[][] matriz = {
    {1, 2},
    {3, 4}
};
System.out.println("Elemento [1][1]: " + matriz[1][1]);  // Accede al valor 4
```

05

Recorridos

1. **Recorrer Arrays Unidimensionales**
   - ➢ **Con bucles for**

```java
int[] numeros = {10, 20, 30, 40};

for (int i = 0; i < numeros.length; i++) {
    System.out.println("Elemento " + i + ": " + numeros[i]);
}
```

   - ➢ **Con bucle for-each**

```java
for (int num : numeros) {
    System.out.println(num);
}
```

05

Recorridos

1. **Recorrer Arrays Multidimensionales**

```java
int[][] matriz = {
    {1, 2},
    {3, 4}
};

for (int i = 0; i < matriz.length; i++) {
    for (int j = 0; j < matriz[i].length; j++) {
        System.out.println("Elemento [" + i + "][" + j + "]: " + matriz[i][j]);
    }
}
```

06

Búsqueda y ordenación

1. Búsquedas en Arrays

Para buscar un elemento en un array, se recorre elemento por elemento comparando con el valor buscado.

```java
int[] numeros = {10, 20, 30, 40};
int buscado = 30;

for (int i = 0; i < numeros.length; i++) {
    if (numeros[i] == buscado) {
        System.out.println("Elemento encontrado en la posición: " + i);
        break;
    }
}
```

06

Búsqueda y ordenación

2. Ordenación de Arrays

Para ordenar un array, usamos el método **Arrays.sort** de la clase java.util.Arrays.

```
import java.util.Arrays;

int[] numeros = {50, 10, 30, 20, 40};
Arrays.sort(numeros);

for (int num : numeros) {
    System.out.println(num);
}
```


PROMETEO