



ORACLE

Academy



Java Foundations

8-1

Matrices unidimensionales

ORACLE
Academy



Objetivos

- En esta lección se abordan los siguientes objetivos:
 - Crear e inicializar matrices unidimensionales
 - Modificar un elemento de matriz
 - Recorrer una matriz unidimensional mediante un bucle for
 - Identificar la causa de un `ArrayIndexOutOfBoundsException`



¿Una variable puede contener más de un valor?

- Hasta el momento hemos utilizado muchos tipos de variables, pero cada variable almacena un valor a la vez:
 - int, string o double
- A continuación se muestra un ejemplo de una variable String, rockband, que puede contener cualquier valor – Joe, Paul, Ed, Rob:
 - Puesto que solo hay 4 valores posibles, cambiar el valor de la variable no es muy difícil

```
String rockBand = "Joe";  
String rockBand = "Paul";  
String rockBand = "Ed";  
String rockBand = "Rob";
```

Número de variables necesarias

- Pero hay ocasiones en las que necesitará que una variable contenga más de un valor
- ¿Qué pasa si quiere definir una variable diferente para cada una de las canciones de RockBand? (¡Significaría que hay 300 variables para cada canción!)
- Sin embargo, crear cientos de variables puede tardar mucho tiempo y ser tedioso

```
String rockBandSong1 = "Rainy day";  
String rockBandSong2 = "Forever";  
String rockBandSong3 = "Something about you";  
String rockBandSong4 = "Love you always";  
.....
```

Las matrices pueden proporcionar una solución

- En Java, una matriz es un contenedor indexado que incluye un juego de valores de un único tipo
- Las matrices permiten crear un único identificador para organizar varios elementos del mismo tipo de dato

0	1	2	3	4	5	6
27	12	82	70	54	1	30

Índices

Elementos

Las matrices pueden proporcionar una solución

- Cada elemento de una matriz se denomina elemento
- Las matrices hacen que almacenar un número elevado de valores y acceder a ellos sea fácil y sencillo

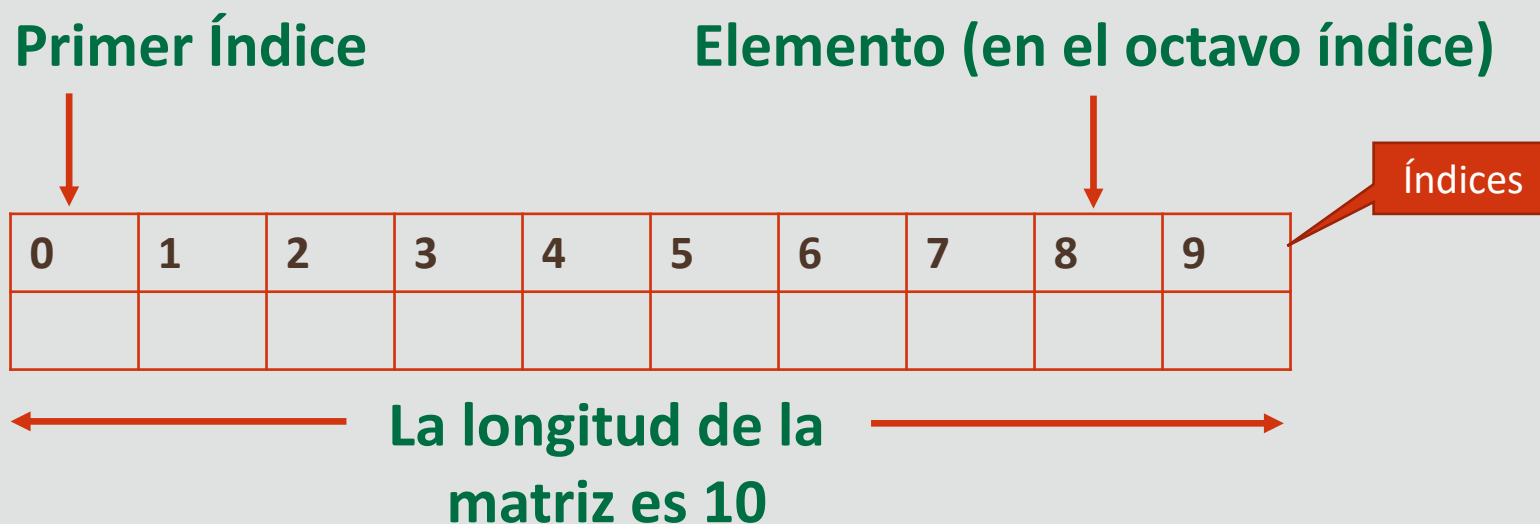
0	1	2	3	4	5	6
27	12	82	70	54	1	30

Índices

Elementos

Se puede acceder a las matrices mediante el índice

- Puede acceder a cada elemento en una matriz mediante su índice numérico
- El índice del primer elemento es 0
- Una matriz de 10 elementos tiene de 0 a 9 índices



Tipos de dato de la matriz

- Las matrices pueden ser de cualquier tipo de dato, pero todos los elementos tienen que compartir el mismo tipo, como:

- Primitivo:

- Ejemplo: matriz de tipos int

27	12	82	70	54	1	30
----	----	----	----	----	---	----

- Objetos predefinidos:

- Ejemplo: Matriz de String

Sun	Mon	Tue	Wed	Thu	Fri	Sat
-----	-----	-----	-----	-----	-----	-----

Tipos de dato de la matriz

- Las matrices pueden ser de cualquier tipo de dato, pero todos los elementos tienen que compartir el mismo tipo, como:
 - Objetos definidos por el programador:
 - (como ejemplos de una clase que creó)
 - Ejemplo: Matriz de objetos de la clase Student

Student1	Student2	Student3	Student4	Student5
----------	----------	----------	----------	----------

Declaración de una Matriz

- Las matrices, como todas las variables, se deben declarar antes de su uso
- Puede declarar una matriz con la sintaxis siguiente:

```
type[] arrayIdentifier;
```

- Observe la notación en corchetes [] después del tipo de dato

Declaración de una matriz de valores de temperatura

- Suponga que desea almacenar diferentes lecturas de temperatura en una matriz
- Puede declarar una matriz de la siguiente manera:

```
double[] temperature;
```

Tipo de dato en función de los elementos que quiere almacenar en la matriz

Subíndice

Nombre de la matriz

Declaración de una matriz: Dos métodos

- Puede declarar una matriz de dos formas:

```
1. int[] prime;  
2. int prime[];
```

- Ambas sintaxis son equivalentes
- El primer formato, en general, es más legible y se debe utilizar

¿Basta con declarar una matriz?

- Declarar una matriz no es suficiente para empezar a utilizarla en el programa
- Antes de utilizar una matriz, debe indicar a Java para que cree espacio en la memoria para los elementos que contendrá

¿Basta con declarar una matriz?

- Utilice la siguiente sintaxis:

```
data_type[] variable_name = new data_type[size];  
variable_name[index] = value; //repeat for each element
```

- El valor de tamaño determina el número de elementos que la matriz puede contener
- Las matrices no pueden crecer por encima de este tamaño

Creación de una matriz

- Por ejemplo, si desea crear una matriz que contenga 100 números enteros, puede realizar las siguientes acciones:

```
int[] myIntArray;  
myIntArray = new int[100];
```

- También puede realizar estas dos líneas en un solo paso:

```
int[] myIntArray = new int[100];
```

¿Qué hacen los fragmentos de código?

```
int[] ages = new int[3];  
ages[0] = 19;  
ages[1] = 42;  
ages[2] = 92;
```

1

```
String[] names = new String[3];  
names[0] = "Mary";  
names[1] = "Bob";  
names[2] = "Carlos";
```

2

Nombre de
Variable

Índice

Valor

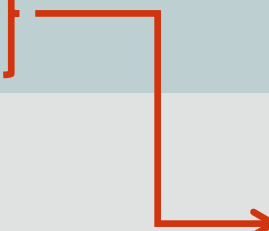
¿Y declarar e inicializar una matriz en un único paso?

- También puede declarar e inicializar la matriz en un solo paso con valores conocidos:

```
type[] arrayIdentifier = {comma-separated list of values};
```

- Por ejemplo, declare las matrices de los tipos String e int:

```
String[] names = {"Mary", "Bob", "Carlos"};  
int[] ages = {25, 27, 48};
```

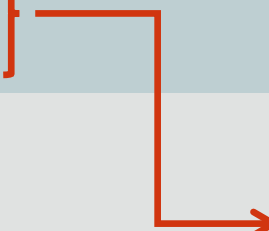


Declaración e
inicialización
en un solo
paso

¿Y declarar e inicializar una matriz en un único paso?

- Observe que este método no especifica el tamaño
- Se asigna un tamaño en función del número de elementos entre llaves ({ })

```
String[] names = {"Mary", "Bob", "Carlos"};  
int[] ages = {25, 27, 48};
```



Declaración e
inicialización
en un solo
paso

Acceso a los elementos de matriz

- Las matrices son estructuras sequenciales, lo que significa que los elementos se almacenan uno después de otro en una matriz
- Puede acceder a un elemento individual de una matriz mediante una notación en corchetes
- Por ejemplo, aquí se muestra cómo puede obtener valores de la matriz ages:

```
int[] ages = {25, 27, 48};  
int myAge = ages[0];  
int yourAge = ages[1];  
System.out.println("My age is " + ages[0]);
```

¿Cómo se define el valor de un elemento de matriz?

- Puede definir los valores para los elementos de la matriz como se muestra a continuación:

```
String[] names = {"Mary", "Bob", "Carlos"};  
names[0] = "Gary";  
names[1] = "Rob";
```

- Después de definir los valores en los elementos en los índices 0 y 1, la matriz names se muestra de la siguiente manera:

0	1	2
Gary	Rob	Carlos
names[0]	names[1]	names[2]

Ejercicio 1

- ¿Puede identificar los tres componentes de una declaración de matriz para cada una de estas matrices de tipos de datos primitivos?
 - Tipo de Dato
 - Nombre
 - Tamaño

```
int[] myArray;
```

```
myArray = new int[20];
```

```
char[] sentence = new char[100];
```


```
double[] teamPoints = new double[5];
```


Inicialización por defecto de matrices

- Cuando las matrices se han declarado pero aún no se han inicializado, a los elementos se les da el valor por defecto asociado con el tipo de dato.
- Por ejemplo:

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

Valores por defecto para los
elementos de esta matriz



Indeks:	0	1	2	3	4
Nilai:	0	0	0	0	0

¿Cómo se accede a la longitud de una matriz?

- Hasta ahora, ha creado una matriz con un número determinado de elementos
- Después de la creación, no puede cambiar la longitud de una matriz. No pueden crecer por encima de este tamaño
- Puede acceder al tamaño de cualquier matriz mediante la propiedad `length` de la matriz

```
int primes[] = {2, 3, 5, 7, 11, 13, 17};  
System.out.println("Array length: " + primes.length);  
  
//prints 7
```

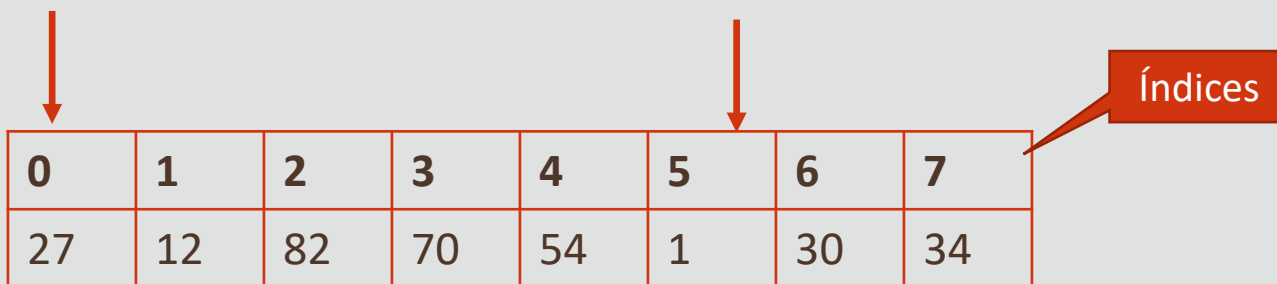
Índices y longitud de matriz

- Por ejemplo, el siguiente fragmento de código muestra el tamaño de la matriz ages:

```
int ages[] = {27, 12, 82, 70, 54, 1, 30, 34};  
System.out.println(ages.length); //prints 8
```

Primer Índice

Quinto Índice



0	1	2	3	4	5	6	7
27	12	82	70	54	1	30	34

La longitud de la
matriz es 8
(ages.length)

Ejercicio 2

- Cree un nuevo proyecto y agréguele el archivo `ArrayEx1.java`
- Examine `ArrayEx1.java`
- Modifique el programa para implantar...
 - Declare una matriz de una dimensión denominada `score` de tipo `int` que pueda contener 9 valores
 - Declare e inicialice una matriz de bytes de una dimensión denominada `values` de tamaño 10, de manera que todas las entradas contengan 1
 - Anule los comentarios de las dos líneas que se han comentado y, a continuación, resuelva los errores de sintaxis

Recorrido de una matriz

- Iterar o recorrer una matriz significa procesar cada elemento de la matriz por el número de índice
- Puede acceder a cada elemento de una matriz para...
 - Imprimir los elementos
 - Buscar un elemento
 - Inicializar los elementos de una matriz con el mismo valor

Uso de un bucle for para recorrer las matrices

- Puede utilizar un bucle for para recorrer las matrices
- El bucle for le permite iterar sobre un rango conocido
- Puede visitar cada elemento de matriz mediante la propiedad de longitud de la matriz en la condición de iteración

```
int[] array = { -20, 19, 1, 5, -1, 27, 19, 5 } ;  
int min = array[0]; // initialize the current minimum  
for (int index=0; index < array.length; index++ )  
    if (array[index] < min)  
        min = array[index] ;  
System.out.println("The minimum of this array is: " + min);
```

¿Cómo se imprimen los valores de una matriz de nombres?

- Considere una matriz de cadenas, names:

```
String names[] = {"Tom", "David", "Mike"};
```

- Recorra la matriz names con el bucle for:

```
for (int idx = 0; idx < names.length; idx++){  
    System.out.println(names[idx]);  
} //end for
```

expresión booleana

El contador se utiliza como
índice de la matriz

Uso de un bucle for-each para recorrer una matriz

- También puede utilizar un bucle for-each, una alternativa al bucle for, para iterar una matriz.
- El bucle for-each...
 - Funciona de la misma forma que el bucle for, pero se implanta de una manera más sencilla.
 - También se le conoce como un bucle for mejorado.

Uso de un bucle for-each para recorrer una matriz

- Sintaxis:

```
for (<type> <iteration variable> : <array name>) {  
    <code_block to be performed for each array element>  
} //end for
```

¿Cómo se imprimen los valores de una matriz de nombres mediante un bucle for-each?

- A continuación se muestra un ejemplo de un recorrido de la matriz de nombres mediante un bucle for-each:

Tipo **Iteración-Variable** **Nombre de la matriz**

```
for(String name: names){  
    System.out.println(name);  
}//end for
```

¿Cómo se imprimen los valores de una matriz de nombres mediante un bucle for-each?

- Para cada iteración del bucle, el siguiente elemento de la matriz se recupera y se almacena en una variable de iteración
- El tipo debe ser el mismo que el de los elementos almacenados en la recolección

Bucle for-each frente a Bucle for

- Bucle for-each

```
for(String name: names){  
    System.out.println(name);  
}//end for
```

- Bucle for

```
for (int idx = 0; idx < names.length; idx++){  
    System.out.println(names[idx]);  
}//end for
```

- La salida de ambos bucles es la misma.

Procesamiento de una matriz de cadenas

El bucle accede a
cada elemento
por turnos

matriz names de los tipos String

George

Jill

Xinyi

Ravi

```
for(String name : names ) {  
    System.out.println("Name is " + name);  
} //end for
```

Cada iteración
devuelve el
siguiente elemento
de la matriz

• Resultado:

```
Name is George  
Name is Jill  
Name is Xinyi  
Name is Ravi
```

Conclusiones

- Veamos un ejemplo en el que hay que...
 - Introducir las puntuaciones de 10 estudiantes mediante un objeto Scanner
 - Mostrar las puntuaciones que ha introducido
 - Calcular el promedio de las puntuaciones que ha introducido

Calculemos la puntuación media

```
public class StudentScores {
    public static void main(String args[]) {
        double scores[] = new double[10];
        double sum = 0.0, avg = 0.0;
        Scanner keyboard = new Scanner(System.in);

        System.out.println("Enter scores of 10 students");
        for(int i = 0; i < scores.length; i++) {
            scores[i] = keyboard.nextInt();
        } //end for
        System.out.println("Display the scores of 10 students");
        for(int i = 0; i < scores.length; i++) {
            System.out.println(scores[i]);
        } //end for
        for(int i = 0; i < scores.length; i++) {
            sum = sum + scores[i];
            avg = sum / scores.length;
        } //end for
        System.out.println("The average score of the class " + avg);
    } //end method main
} //end class StudentScores
```

Ejercicio 3

- Agregue el archivo `ComputeAvg.java` al proyecto creado para el ejercicio 2
- Examine `ComputeAvg.java`
- Modifique el programa para implantar...
 - En una clase determinada, hay cinco pruebas, cada una con un valor de 100 puntos
 - Introduzca cinco puntuaciones de las pruebas de la consola
 - Almacene las puntuaciones de las pruebas en una matriz
 - Calcule las puntuaciones medias de los estudiantes

¿Qué es ArrayIndexOutOfBoundsException?

- Como ya sabe, una matriz tiene un tamaño fijo
- El índice debe estar en un intervalo de rango $[0, n-1]$, en que n es el tamaño de la matriz
- Si un índice es negativo o mayor o igual a tamaño de la matriz, el índice de matriz está fuera de los límites
- Si un índice de matriz está fuera de los límites, JVM devuelve ArrayIndexOutOfBoundsException
- Esto se denomina comprobación de límites automática

¿Qué pasa cuando se produce esta excepción?

- Se ha devuelto `ArrayIndexOutOfBoundsException` solo en tiempo de ejecución
- El compilador Java no comprueba esta excepción cuando se está compilando un programa
- El programa termina si esta excepción no se ha manejado

¿Cómo se Identifica ArrayIndexOutOfBoundsException?

```
public static void main(String[] args) {  
    int primes[] = {2, 3, 5, 7, 11, 13, 17};  
    System.out.println("Array length: " + primes.length);  
    primes[10] = 20; //  
  
    System.out.println("The first few prime numbers are:");  
    for (int i : primes) {  
        System.out.println(i);  
    }//end for  
}//end method main
```

El índice de la matriz es
de 0 a 6 y está intentando
acceder a un elemento
en el índice 10

• Resultado:

```
Array length: 7  
Exception in thread "main"  
java.lang.ArrayIndexOutOfBoundsException: 10  
    at arraysdemo.ArraysDemo.main(ArraysDemo.java:21)  
Java Result: 1
```

Ejercicio 4

- Agregue el archivo `ArrayEx2.java` al proyecto creado para el ejercicio 2
- Examine `ArrayEx2.java`
- Realice lo siguiente:
 - Ejecute el proyecto y observe el error
 - Modifique el programa para resolver el error
 - Mediante un bucle `for-each`, muestre todos los exploradores que se almacenan en la matriz

Resumen

- En esta lección, debe haber aprendido lo siguiente:
 - Crear e inicializar matrices unidimensionales
 - Modificar un elemento de matriz
 - Recorrer una matriz unidimensional mediante un bucle for
 - Identificar la causa de un `ArrayIndexOutOfBoundsException`





ORACLE

Academy

