Java para Cibernética y Computación

Colegio de Ciencias y Humanidades – Plantel Oriente

Instructores:

¡¡¡TODOS LOS PROFESORES INSCRITOS EN EL CURSO!!!

Día 2

API de Java

- https://docs.oracle.com/javase/10/docs/api/index.html?overview-summary.html

Strings

Cadenas

Strings

- Ante todo dejar claro que los Strings no son específicamente tipos primitivos en Java. También son una clase implementada con ciertas funcionalidades que hacen que su uso sea comparable al de los tipos primitivos en ciertos aspectos.

- Formas de declarar una cadena:

```
String nombre;
String nombre = “”;
String nombre = “Juan”;
```

Algunos métodos de las cadenas

- equals() `//Sirve para comparar cadenas`
- length() `//Devuelve el tamaño de la cadena`
- chartAt()`//Devuelve el caracter indicado en el parámetro`
- compareTo() `//Compara dos cadenas según la tabla ASCII`
- compareToIgnoreCase()`//Compara dos cadenas sin importar mayúsculas o minúsculas`
- concat()`//Concatena dos cadenas`
- replace()`//Devuelve la cadena cambiando el caracter que se pasa como parámetro`
- substring`()//Corta una cadena de una posición inicial a una final`
- toUpperCase()`//Convierte la cadena a mayúsculas`
- toLowerCase()`//Convierte la cadena a minúsculas`

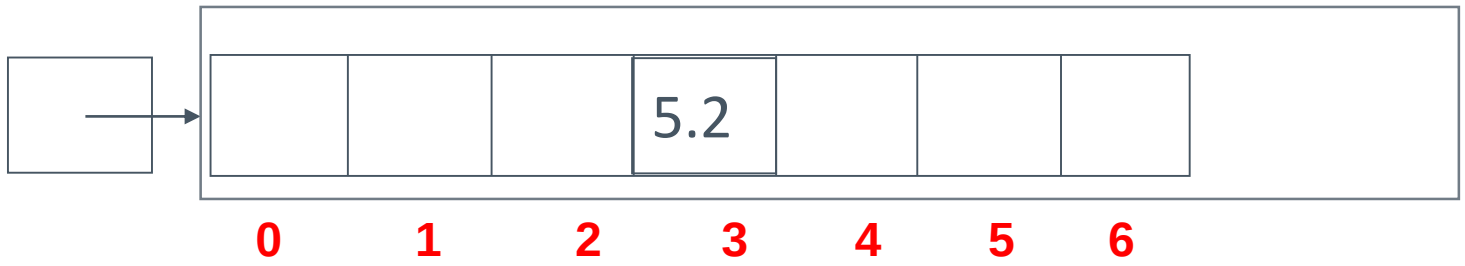
Arrays

Arreglos

¿Qué es un arreglo?

› Un arreglo es un objeto que almacena datos del mismo tipo.

› Los más comunes son los arreglos de una dimensión (vectores) y de dos dimensiones (matrices).



Declaración de un arreglo

› La estructura de declaración de un arreglo es la siguiente:

```
tipo_de_dato[] nombre_variable;
```

Ejemplos:

```
char[] arreglo;
int[] arreglo;
double[] arreglo;
String[] arreglo;
```

Asignar tamaño a un arreglo

- Al ser un objeto en el que se guardan varios valores de un tipo de dato en específico, a los arreglos se les debe asignar un tamaño y se hace de la siguiente forma:

```
arreglo = new tipo_de_dato[tamaño];
```

Ejemplos:

```
arreglo = new char[5];
arreglo = new int[10];
arreglo = new double[4];
arreglo = new String[20];
```

Inicializar un arreglo

› Se puede hacer por separado:

```
int[] arreglo = new int[3];
      arreglo[0] = 17;
      arreglo[1] = 20;
      arreglo[2] = 18;
```

› En una sola línea:

```
int[] arreglo = {17,20,18};
```

Acceder a un elemento del arreglo

- Se puede acceder a cada elemento de un arreglo de forma separada:

```
//Imprime el elemento del arreglo ubicado en la posición 1
System.out.println(arreglo[1]);
```

- Imprimir todos los elementos del arreglo de forma dinámica:

```
for(int i = 0; i < 5; i++){
   System.out.println(arreglo[i]);
}
```

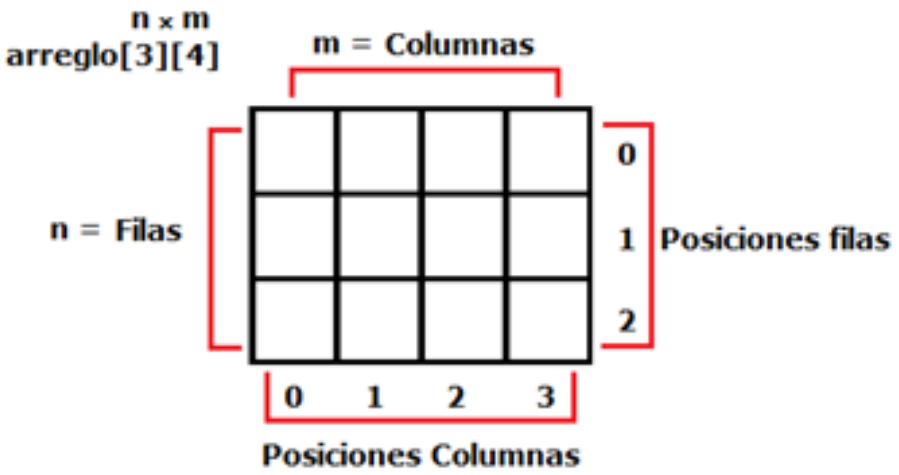
Tamaño de un arreglo

› Para obtener el tamaño de un arreglo se usa el atributo **length**

```
arreglo.length
```

Arreglo bidimensional (Matriz)

› Este tipo de arreglos son conocidos como matrices y corresponden a una estructura de datos que puede almacenar muchos más datos que los arreglos unidimensionales, ya que se componen de n filas por m columnas.



Declaración e inicialización

› La estructura de declaración de un arreglo bidimensional es la siguiente:

```
tipo_de_dato[][] nombre_variable;
nombre_variable = new tipo_de_dato[filas][columnas];
```

```
Ejemplo:
int[][] matrizDeEnteros;
matrizDeEnteros = new int[3] [4];
```

Asignar valores

Llenado fila 0 columnas del 0 al 3

```
matrizDeEnteros[0][0]=1;
matrizDeEnteros[0][1]=3;
matrizDeEnteros[0][2]=5;
matrizDeEnteros[0][3]=7;
```

Llenado fila 1 columnas del 0 al 3

```
matrizDeEnteros[1][0]=5;
matrizDeEnteros[1][1]=4;
matrizDeEnteros[1 ][2]=1;
matrizDeEnteros[1][3]=16;
```

Llenado fila 2 columnas del 0 al 3

```
matrizDeEnteros[2][0]=7;
matrizDeEnteros[2][1]=9;
matrizDeEnteros[2][2]=61;
matrizDeEnteros[2][3]=13;
```

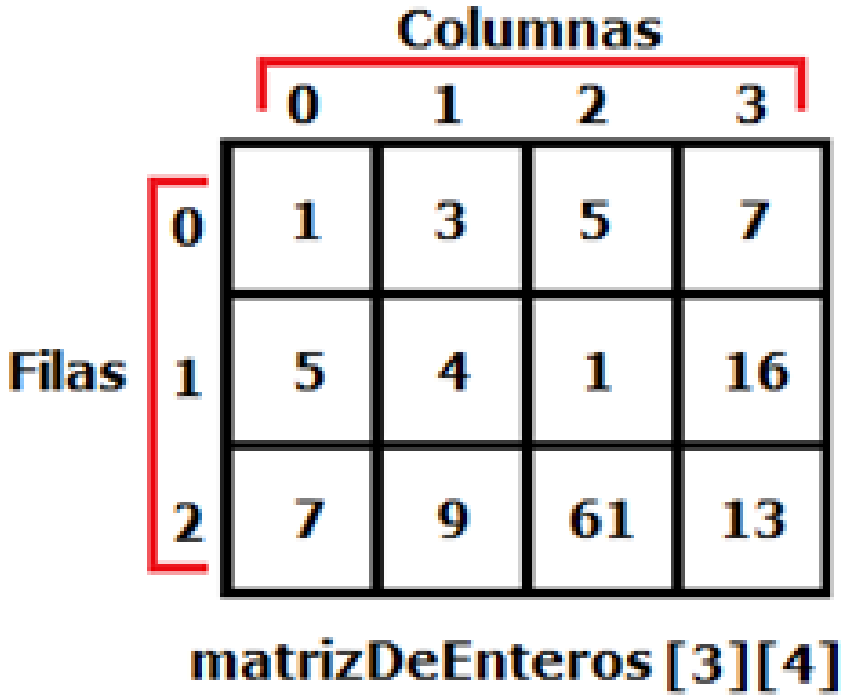
Columnas

Filas	0	1	2	3
0	1	3	5	7
1	5	4	1	16
2	7	9	61	13

matrizDeEnteros [3][4]

Otra forma de asignar valores

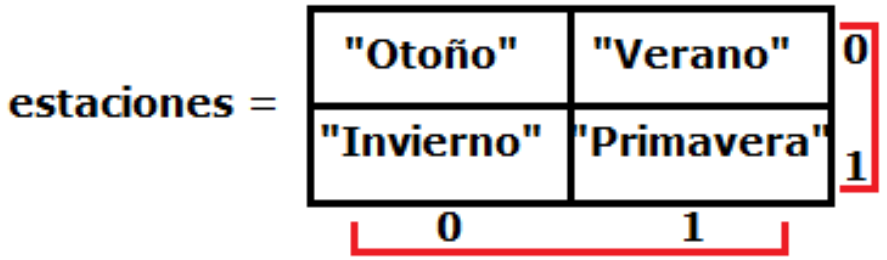
```
matrizDeEnteros = { {1, 3, 5, 7} , {5, 4, 1, 16} , {7, 9, 61, 13} };
```



Matriz de Strings

```
14  public static void main(String[] args) {
15
16      String estaciones[][]=new String[2][2];
17
18      //llenado de la matriz
19      estaciones[0][0]="Otoño";
20      estaciones[0][1]="Verano";
21      estaciones[1][0]="Invierno";
22      estaciones[1][1]="Primavera";
23
24      //Obteniendo información de la matriz
25      System.out.println("estación en la posición (0,0): "+estaciones[0][0]);
26      System.out.println("estación en la posición (0,1): "+estaciones[0][1]);
27      System.out.println("estación en la posición (1,0): "+estaciones[1][0]);
28      System.out.println("estación en la posición (1,1): "+estaciones[1][1]);
29  }
```

Lo anterior crea una matriz de tipo String y tamaño 2x2 con posiciones de 0 a 1 en filas y 0 a 1 en columnas



ArrayList

ArrayList

- La clase ArrayList en Java, es una clase que permite almacenar datos en memoria de forma similar a los Arrays, con la ventaja de que el numero de elementos que almacena, lo hace de forma dinámica, es decir, que no es necesario declarar su tamaño como pasa con los Arrays.
- Los ArrayList nos permiten añadir, eliminar y modificar elementos de forma trasparente para el programador.

Declaración de un ArrayList

- `ArrayList<tipo> nombreArray = new ArrayList <tipo>();`

- Ejemplos:
- `ArrayList<String> nombreArrayList = new ArrayList<String>();`
- `ArrayList<Integer> nombreArrayList = new ArrayList<Integer>();`
- `ArrayList<Float> nombreArrayList = new ArrayList<Float>();`
- `ArrayList<Double> nombreArrayList = new ArrayList<Double>();`

Algunos de métodos de los arrayList

- **add()**//Añade un elemento
- **size()**//Devuelve el número de elementos del arreglo
- **get()**//Extrae el elemento del arreglo que se le pasa como parámetro
- **remove()**//Borra elementos del arreglo
- **clear()**//Borra todos los elementos del arreglo
- **isEmpty()**//Devuelve True si el ArrayList esta vacío. Si no devuelve False.
- **clone()**//copia en contenido de un arreglo a otro