Argentina programa 4.0

Arrays y Strings

Sintaxis básica

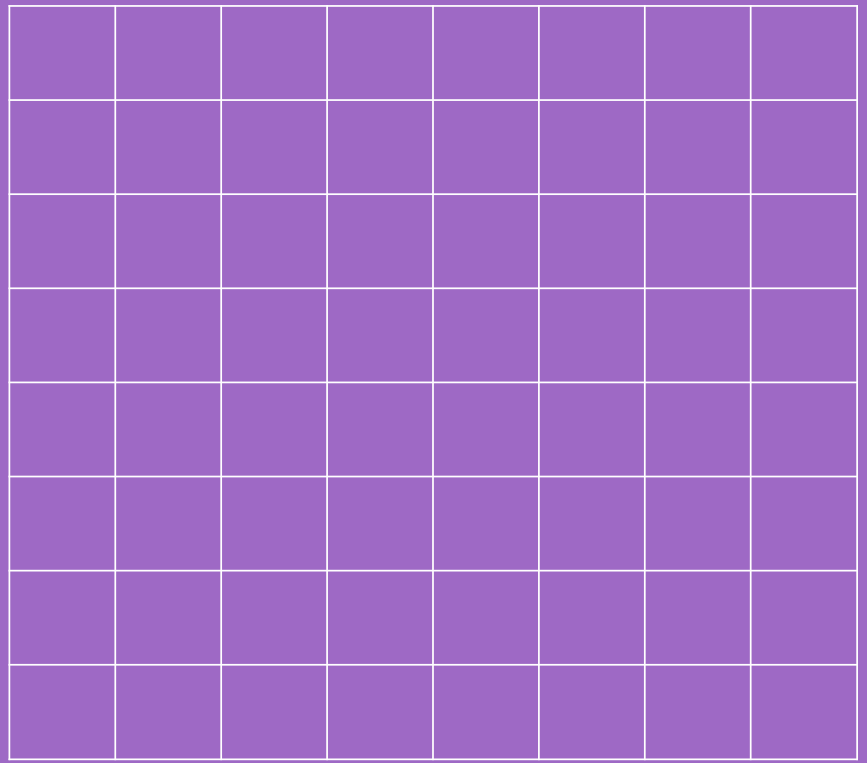
“Desarrollador Java Inicial”

Agenda

Argentina programa 4.0

- Arrays
  - Definición
  - Declaración
  - Iteración
- Strings
  - Declaración
  - Uso
  - Métodos principales

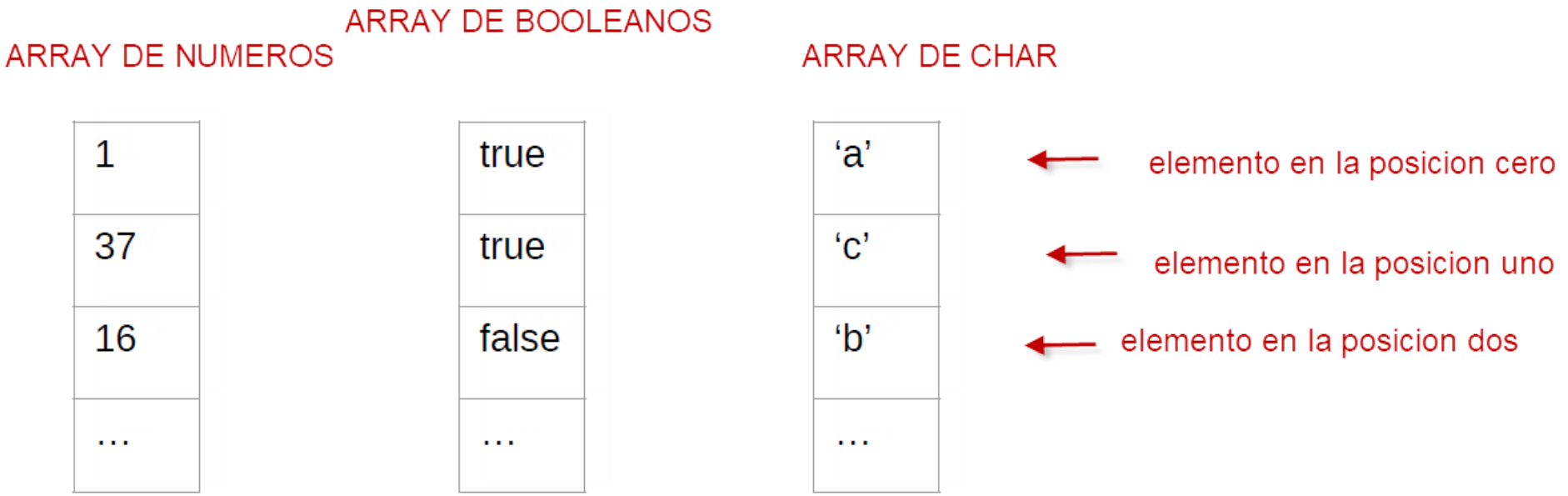
Arrays



Definición

Argentina programa 4.0

Es muy importante en todo algoritmo representar un grupo de elementos. La forma más simple de hacerlo en todo lenguaje son los *Arrays* o “Arreglos”. En particular en Java, los arrays son estructuras de datos que agrupan un conjunto de elementos del mismo tipo:



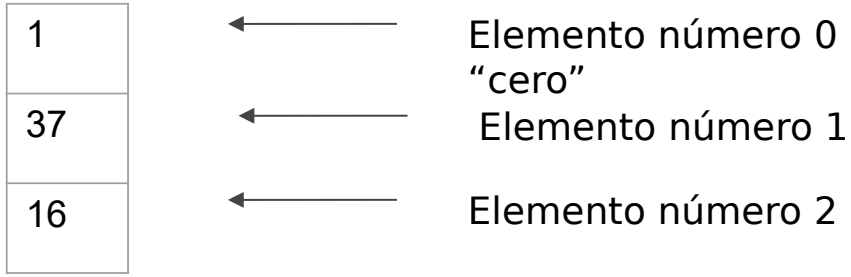
Ejemplo de array de números

```
int numeros[] = new int[3]; // Así se declara un arreglo de enteros
```

se declara la variable

se crea el *array*

```
numeros[0] = 1; // asignación
numeros[1] = 37;
numeros[2] = 16;
```

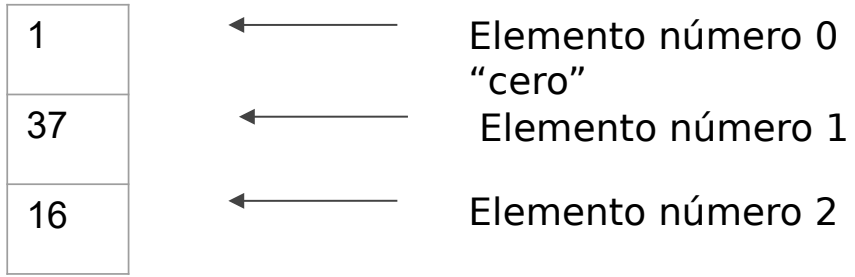



```
// También puede inicializarse directo con valores
int numeros[] = new int[]{1,37,16};
```

Ejemplo de array de números - con

Argentina programa 4.0

```
int numeros[] = new int[3]; // Así se declara un arreglo de enteros

numeros[0] = 1;
numeros[1] = 37;
numeros[2] = 16;
```




```
System.out.println(numeros.length)
// imprime por pantalla: 3
// La sintaxis de punto “.” indica que estamos accediendo a una propiedad de
la variable. Lo que vimos como “tipos primitivos” no lo permiten, pero los
arrays tienen una fundamental: “length”, que nos dice el tamaño del array.
for(int indice=0;indice<numeros.length;indice++){
     System.out.println(numeros[indice]);// imprime cada elemento
}
```

Java - Sintaxis Básica - Vectores/Arrays básicos

Argentina programa 4.0

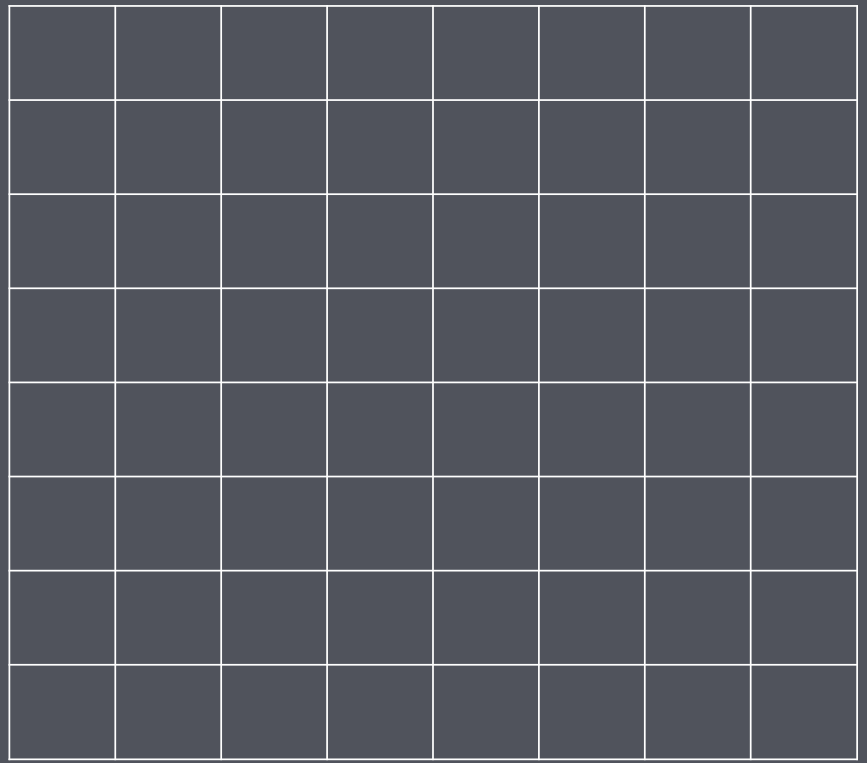
```java
int numeros[] = new int[10];
char letras[] = {'a','b','c'};

numeros[0] =  10;
numeros[5] =  14;

System.out.println(numeros[5] / 7);
//2
for (char letra : letras){
    System.out.println(letra);
}
// a
// b
// c
```

Java - Vectores/Arrays básicos - Importante

Argentina programa 4.0

- Siempre están numerados desde cero, pero la propiedad “length“ es el tamaño total del vector. Por ejemplo, si hay un array de 5 elementos, la propiedad length es 5, pero el índice para acceder al último elemento es el “4”.
- Los arrays son de tamaño fijo. Es decir, si quiero almacenar más elementos, debo crear un nuevo Array
- Para tener un conjunto de elementos variable, vamos a ver colecciones más adelante.

Strings



String

Argentina programa 4.0

- String es el tipo de dato (Clase en realidad, ya veremos de qué se trata) de Java Para representar cadenas de texto, de cualquier tamaño.
- Todos los lenguajes de programación tienen este dato
- Los Strings se declaran con comillas dobles

```
String txt1 = "Hola!";
String txt2 = "Chau!";
```

- Se puede pensar como un array del tipo de dato “char”, aunque como veremos, tiene muchas operaciones particulares. Por ejemplo para acceder a un carácter en particular se puede hacer:

```
txt1.charAt(2); // "l"
```

String - Operaciones

Argentina programa 4.0

- Existen numerosas operaciones que se pueden hacer con los strings, vamos a tomar las variables de ejemplo:

```
String txt1 = "Hola!";
String txt2 = "Chau!";
```

- Concatenar Strings con “+”

```
String txt3 = txt1 + txt2 ;         // "Hola!Chau!"
String txt4 = txt1 + " " + txt2 ; // "Hola! Chau!"
```

- Comparar

```
txt1.equals(txt2); // false
txt1.equals(txt1); // true
```

-> notar que NO debe usarse el operador == para comparar Strings

- Largo de un String

```
txt1.length(); // 5
```

String - Operaciones

```
String unTexto = "laLA";
```

- Pasar a mayúscula o minúscula

```
unTexto.toLowerCase() + unTexto.toUpperCase(); // lalaLALA
```

- Indicar si contiene otro string

```
unTexto.contains("la"); //true
unTexto.contains("aL"); //true
unTexto.contains("La"); //false
```

- Ver si inicia o termina con otra cadena

```
unTexto.startsWith("la"); // true
unTexto.endsWith("LE"); // false
```

- Reemplazar

```
unTexto.replace("la", "le"); // leLA
```

Es importante notar que toLowerCase y replace, NO cambian el valor de la variable "unTexto", sino que retornan un nuevo String, que puedo guardar en otra variable o imprimir por pantalla. Por ejemplo se pueden encadenar:

```
"hola que tal?".replace(
"hola","chau").toUpperCase();
// CHAU QUE TAL?
```

String - Operaciones - Split

Argentina programa 4.0

Quizás la operación más usada de String, es “split” el cual convierte un String en un String[] (es decir un array de strings), a partir de un separador. Por ejemplo:

```
String saludo = "hola que tal?";

String[] saludoPartido1 = saludo.split(" ");

// "hola", "que", "tal?"

String[] saludoPartido2 = saludo.split("a");

// "hol", " que t", "l?" --> notar la presencia de espacios
```

String - Escape y caracteres especiales

Argentina programa 4.0

Todos los lenguajes de programación cuentan con caracteres “especiales”, ya que hay cosas que o bien no se pueden escribir, coinciden con palabras reservadas o la sintaxis no las soporta. En el caso de Java, el caracter especial es la contrabarra “\” (la barra que no es la de dividido). Vamos con unos ejemplos:

- Enter / Cambio de línea

```
"hola que tal?\n" → \n es la forma de hacer un cambio de línea, ya que java no permite strings multi línea
```

- Tabulación (no es lo mismo que varios espacios)

```
"Pedro\t18\tprogramador\n" // \t escribe una tabulación
```

- Escribir una comilla doble: si con “ declaramos un string, como hacemos para que aparezca en el medio del texto que queremos escribir?

```
"hola, \"que tal\" " // con \ decimos que “escapamos” el caracter
```

- Escribir una contrabarra “\”

```
"Esto es una contra barra \\ " // con \ escapamos a la \ misma, si lo imprimimos esto dice:
                                                    Esto es una contrabarra \
```

String - Importante para más adelante

Argentina programa 4.0

- Para tener una referencia completa de la cantidad de cosas que se pueden hacer con String, y de paso para ir rompiendo el hielo, puede consultar el JavaDoc de la misma. Los Javadocs son la forma en la cual se documenta la referencia del código, sobre todo de las clases base de Java.

Class String

java.lang.Object
    java.lang.String

All Implemented Interfaces:

`Serializable, CharSequence, Comparable<String>`

```
public final class String
extends Object
implements Serializable, Comparable<String>, CharSequence
```

The `String` class represents character strings. All string literals in Java programs, such as `"abc"`, are implemented as instances of this class.

Argentina
programa
4.0

Gracias!