

Unidad 4

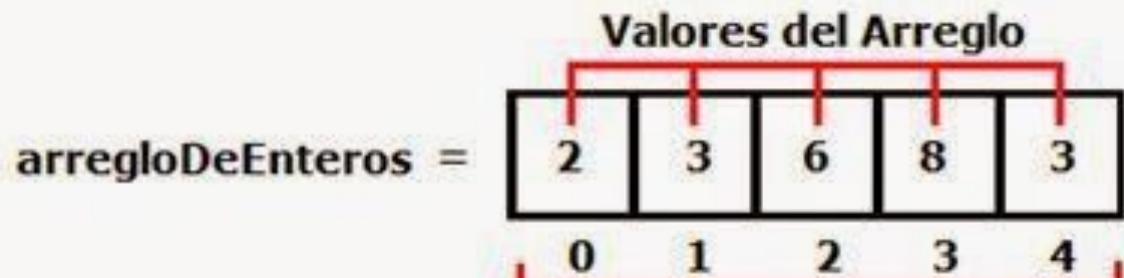
Arreglos y ArrayList

Arreglo

Un arreglo es un objeto que contiene un conjunto de variables del mismo tipo en su interior.

La cantidad de elementos que tiene un arreglo es estática, fija y se define en el momento de su declaración y no se puede modificar.

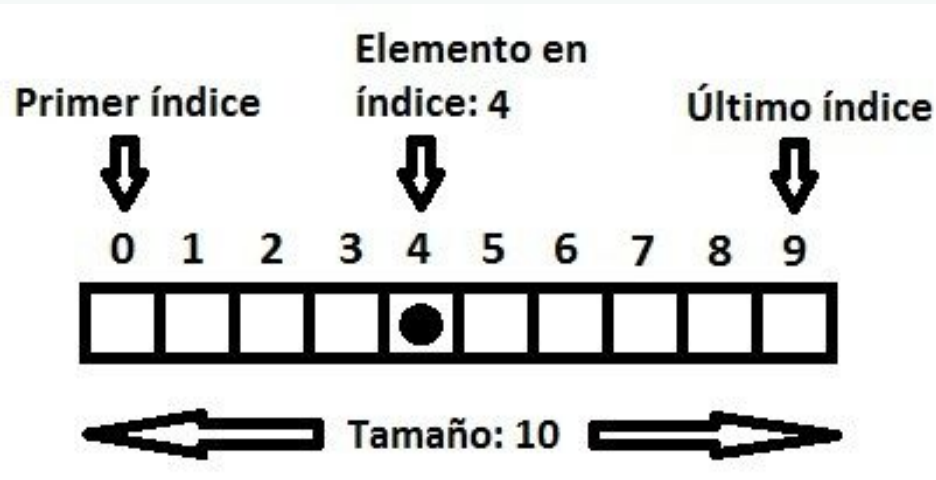
El índice del primer elemento es el cero y el índice del último es la (cantidad de elementos - 1).



Declaración de un Arreglo

Un arreglo se puede declarar de las siguientes formas:

- `int a[];`
- `String []cadena;`
- `double[] ejemplo;`
- `int ejemplo[]=new int[10];`



Inicialización de un Arreglo

Formas de inicializar un arreglo

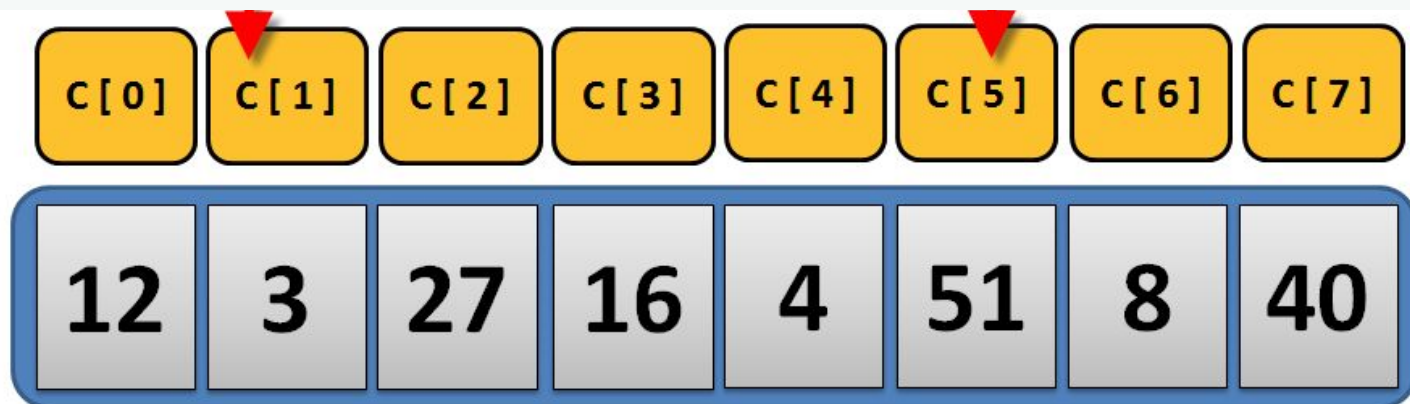
Forma 1: `int c[]={12,3,27,16,4,51,8,40};`

Forma 2: `int c2[]=new int[8];`

`c2[0]=12;`

`c2[1]=3;`

`c2[2]=27;`



Acceso a los Elementos de un Arreglo

Se puede acceder a un elemento de un arreglo en específico indicando su índice dentro de unos corchetes, si se desea imprimir cada valor se puede utilizar un ciclo de iteración (while/for)

```
int c[]={12,3,27,16,4,51,8,40};  
System.out.println(c[2]); //imprime 27  
//Ciclo para imprimir todos los valores  
for(int i=0;i<c.length;i++){  
    System.out.println(c[i]);  
}
```



Ciclo for Mejorado

En Java existe una versión mejorada del ciclo for, conocida como for-each en otros lenguajes, que permite una mayor accesibilidad a los elementos de un arreglo ó conjunto.

```
String paises[]={“México”, “Canadá”, “Rusia”};  
for(String pais : paises){  
    System.out.println(pais);  
}
```



Ejemplo 1

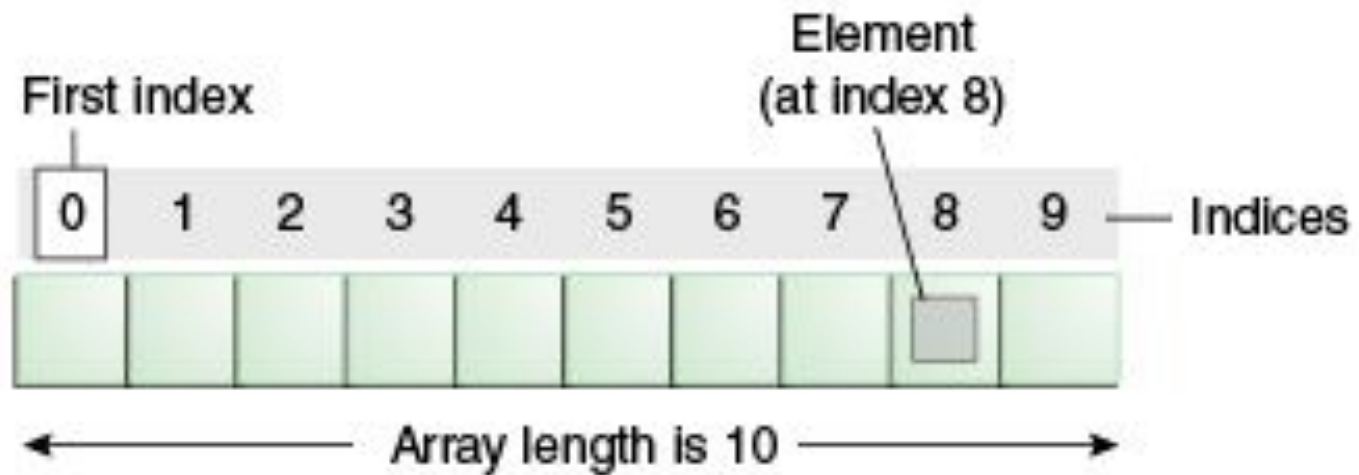
Realizar un programa en el que el usuario ingrese un numero del 1 al 12 e imprimir en pantalla el mes correspondiente a dicho número.

1.	ENERO	7.	JULIO
2.	FEBRERO	8.	AGOSTO
3.	MARZO	9.	SEPTIEMBRE
4.	ABRIL	10.	OCTUBRE
5.	MAYO	11.	NOVIEMBRE
6.	JUNIO	12.	DICIEMBRE



Ejercicio 1

Desarrollar un programa que le pregunte al usuario una serie de diez números, guardarla en un arreglo e imprimir dicho arreglo en el orden en el que el usuario los ingreso y en orden inverso.



Paso de Arreglos a Métodos

Un arreglo se puede pasar a un método como un parámetro, al hacerlo cualquier cambio al parámetro afecta a la variable original.

```
void funcion(int[] arreglo){  
    for(int i = 0; i < arreglo.length; i++){  
        arreglo[i] = 0;  
    }  
}
```

Esta función inicializa el arreglo en cero



Ejemplo 2

Realizar una función que permite efectuar la suma de dos arreglos, guardando la suma en el primero arreglo y regresando true o false si la operación se pudo realizar.

Argumentos de longitud variable

Existe una forma para que se le pueda pasar a una función una serie elementos (ilimitados) del mismo tipo.

En la función hay que indicar que el argumento de longitud variable, se realiza de la siguiente forma:

```
void funcion(String... ejemplo){  
    ...  
}
```



Argumento de Longitud Variable

Para pasar varios elementos se realiza de la siguiente forma:

```
funcion("cadena1", "cadena2", "cadena3");
```

el parámetro String... ejemplo podemos trabajarlo como un arreglo String ejemplo[]



Ejemplo 3

Realizar una función que reciba n números y regrese el promedio de los mismos.

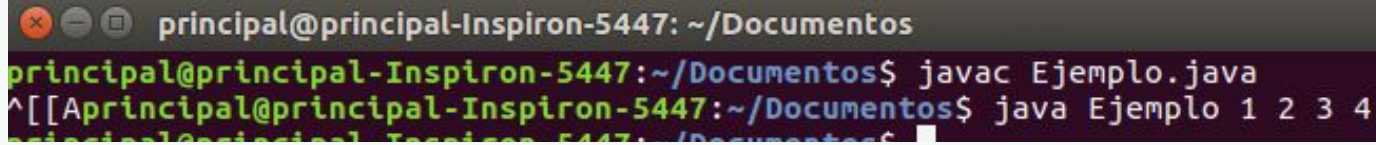
Ejercicio 2

Revisar si una palabra (string) está en mayúsculas, en ese caso regresar true y en caso contrario false



Argumentos desde Línea de Comandos

Cuando se ejecuta un programa desde línea de comandos se puede pasar argumentos, de la siguiente manera:



```
principal@principal-Inspiron-5447: ~/Documentos
principal@principal-Inspiron-5447:~/Documentos$ javac Ejemplo.java
^[[Aprincipal@principal-Inspiron-5447:~/Documentos$ java Ejemplo 1 2 3 4
```

Para poder usar dichos argumentos utilizamos la variable args del método main:

```
public static void main(String[] args) {
    for(String parametro : args){
        System.out.println(parametro);
    }
}
```



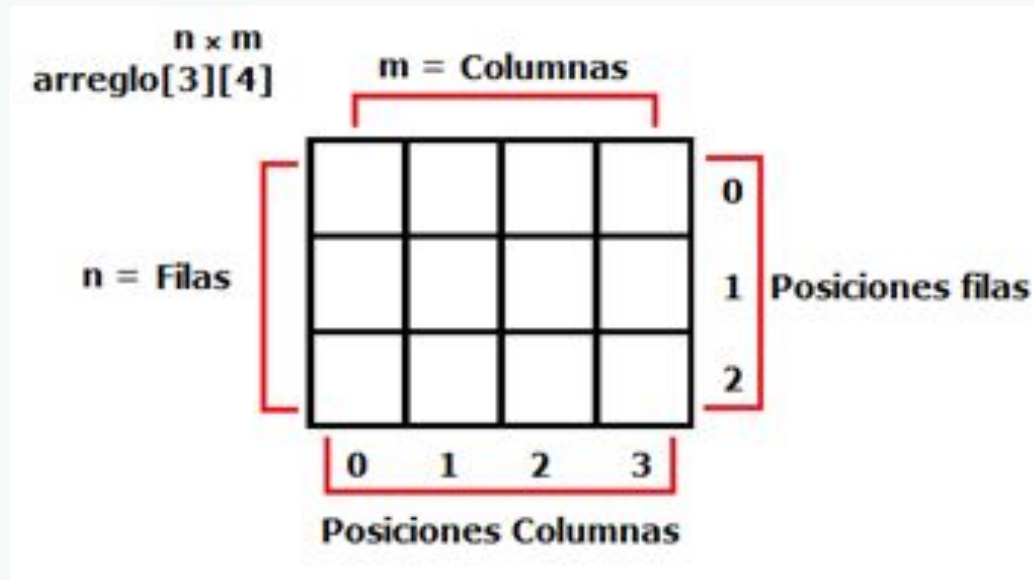
Arreglo Bidimensional (Matriz)

Un arreglo bidimensional es un arreglo cuyo tipo de datos es otro arreglo.

La forma de declarar un arreglo de este tipo es:

```
int arregloBi[][]=new int[filas][columnas];
```

```
int arregloBi[][]={{1,2,3},{4,5,6}};
```



Acceso a los Elementos de un Arreglo Bidimensional

Para acceder a los elementos de un arreglo bidimensional hay que indicar el número de fila y columna al que queremos acceder.

```
int arregloBi[][]={{1,2,3},{4,5,6}};  
System.out.println(arregloBi[1][2]); //imprime 6  
//para imprimir todos en pantalla  
for(int i=0;i<arregloBi.length;i++){ //filas  
    for(int j=0;j<arregloBi[i].length;j++){ //columnas  
        System.out.println(arregloBi[i][j]);  
    }  
}
```



Arreglo Multidimensional

Un arreglo multidimensional está formado de más de dos dimensiones, se declara de una forma parecida al bidimensional:

```
int arregloMulti[] [] [];
```

se inicializa, nuevamente, de una forma parecida al bidimensional:

```
int arregloMulti[] [] []=new int[num1][num2][num3];
```



Arreglos Multidimensionales

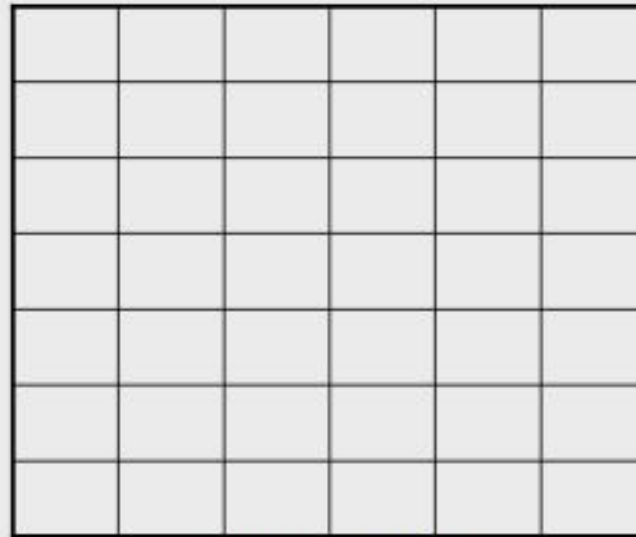
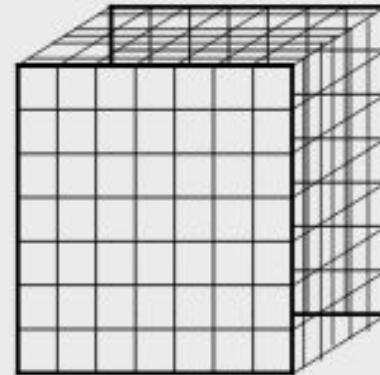
Unidimensional

`int a[20];`



Tridimensional

`float a[7][7][4];`



Bidimensional

`Int a[6][7];`



Ejemplo 2

Realizar un programa en el que exista una clase llamada matriz que contenga:

1. Las operaciones suma, multiplicación por otra matriz, multiplicación por un escalar,
2. Que se imprima en pantalla
3. Que dicha clase contenga un método para obtener una matriz identidad dado un numero (método estático)



Realizar un programa que determine si una matriz dada es cuadrada (si la matriz es $n \times n$ regresar True, y regresar False en caso contrario).

De ser cuadrada, determinar también si es una matriz identidad e imprimir en pantalla.

Incluir un método para obtener la traza de dicha matriz identidad.



Clase Arrays

Existe una clase llamada Arrays que contiene métodos estáticos que permiten trabajar con arreglos de una forma más sencilla. Algunos de estos métodos son:

```
static int binarySearch(tipo nombre[], tipo datoABuscar);  
static boolean equals(tipo arreglo1 [], tipo arreglo2 []);  
static void fill(tipo arreglo[], tipo valor);  
static void sort(tipo arreglo[]);  
static tipo[] clone(tipo arreglo[]);
```



Ejemplo 3

Realizar un programa donde se muestre el uso de las principales funciones de la clase Arrays

One Dimensional array

Initialization `int a[] = new int [12];`

Value	1	2	3	4	5	6	7	8	9	10	11	12
Index	a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]	a[9]	a[10]	a[11]

`System.out.print(a[5]);`

Output: 6



ArrayList

La clase ArrayList permite mantener en memoria una colección dinámica de objetos. A diferencia de un arreglo normal, en un ArrayList puede cambiar la cantidad de elementos que posee en tiempo de ejecución.

Para poder usar la clase hay que importarla de:

`java.util.ArrayList`

Se declara de la forma:

```
ArrayList<Clase> nombre=new ArrayList<Clase>();
```



Métodos de la Clase ArrayList

Métodos NO estáticos importantes:

`boolean add(Tipo elemento);`

`void add(int indice, Tipo elemento);`

`void clear();`

`boolean contains(Object objeto);`

`tipo get(int indice);`

`boolean isEmpty();`

`int remove(Object objeto);`

`boolean remove(int indice);`

`int size();`

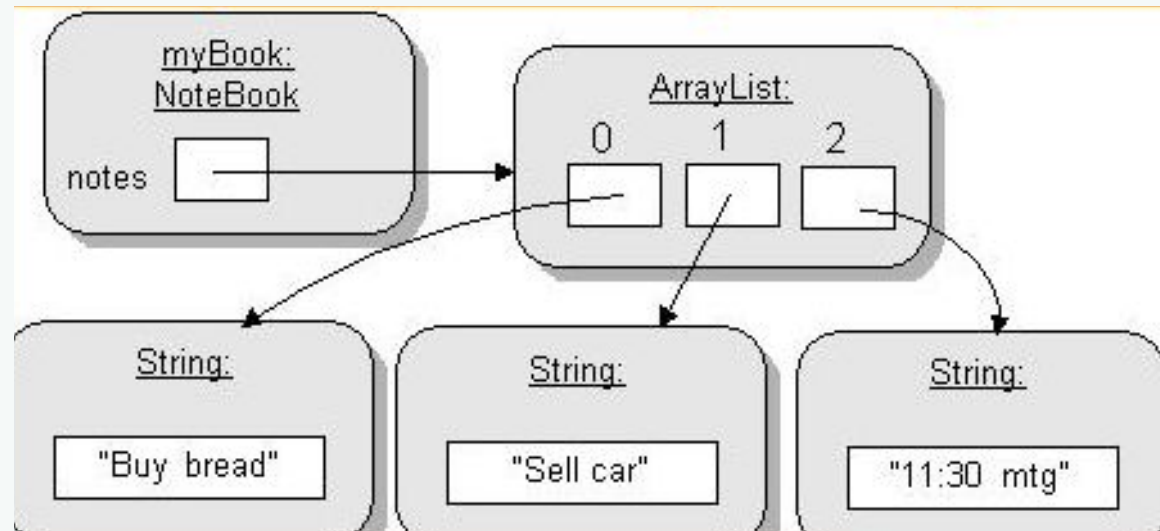
Documentación completa:

<https://docs.oracle.com/javase/8/docs/api/java/util/ArrayList.html>



Ejemplo 4

Realizar un programa para demostrar el uso de un arrayList



Ejercicio 2

Hacer un programa que realice un cifrado César con un desplazamiento de 3 caracteres.

Recibe como argumentos un 1 si se quiere cifrar o un 0 si se quiere descifrar por línea de comandos. Ejemplo: `java Cifrado 1`
Dependiendo de la bandera se realizará el desplazamiento de 3 caracteres ya sea a la derecha o a la izquierda. Ejemplo:

`java Cifrado 1`

Cadena ingresada → hola

Cadena cifrada → krñd

`java Cifrado 0`

Cadena ingresada → dglrv

Cadena descifrada → adiós

.

