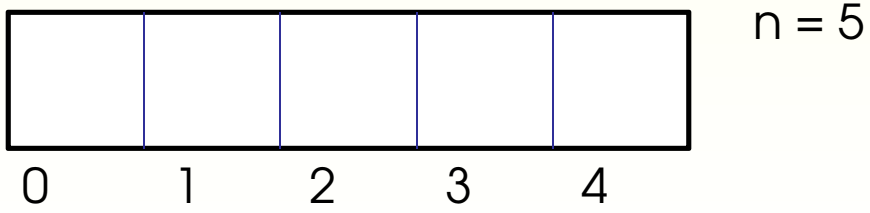
Programación orientada a objetos

Arrays

Dr. (c) Noé Alejandro Castro Sánchez

Arrays

- Medio para almacenar elementos (datos primitivos u objetos) del mismo tipo de dato.
- Cada elemento es accesible mediante un número entero llamado índice
- El primer elemento corresponde al índice 0
  - El último será *n – 1*



Arrays (II)

- Se pueden crear arreglos de cualquier tipo de dato
- Se inicializan por defecto del tipo indicado
- Se puede acceder al número de elementos que puede almacenar con la variable miembro (de instancia) *length*

```
miarreglo.length;
```

Declaración

- Se pueden crear arreglos de cualquier tipo de dato

```
int[] calif;
char[] s;
Point[] p; // Point representa una clase
Dog[] d;   // Dog representa una clase
```

- No se crean los objetos mismos (sólo referencias)

Creación

- Se usa la palabra reservada **new**

```
int[] calif; // declaración
Dog[] dogg;  // declaración

calif = new int[25]; // creación
dogg  = new Dog[3];

int[] linea = new int[15];
```

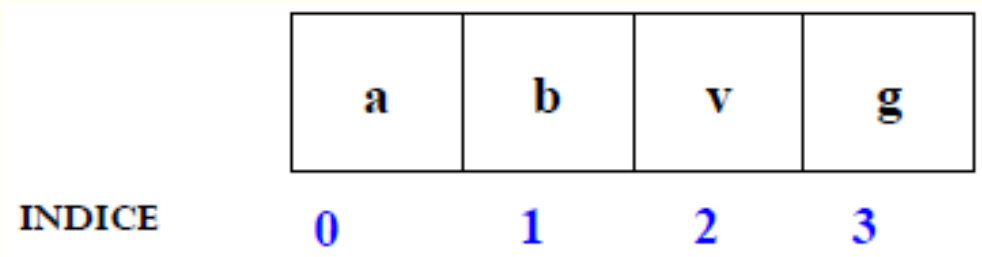
Inicialización

```
calif = new int[25]; // creación
calif[0]=10; // incialización…
calif[1]=9;
calif[2]=10;

d = new Dog[3]; // creación
d[0] = new Dog(); // incialización…
d[1] = new Dog();
d[2] = new Dog();
```

Inicialización (II)

- Se pueden inicializar al momento de la declaración (entre llaves)

```
char[] letras = {‘a’, ‘b’, ‘v’, ‘g’};
Dog[] ds = {new Dog(), null, null};
```



Asignación entre arreglos

- Es posible asignar un arreglo a otro arreglo:

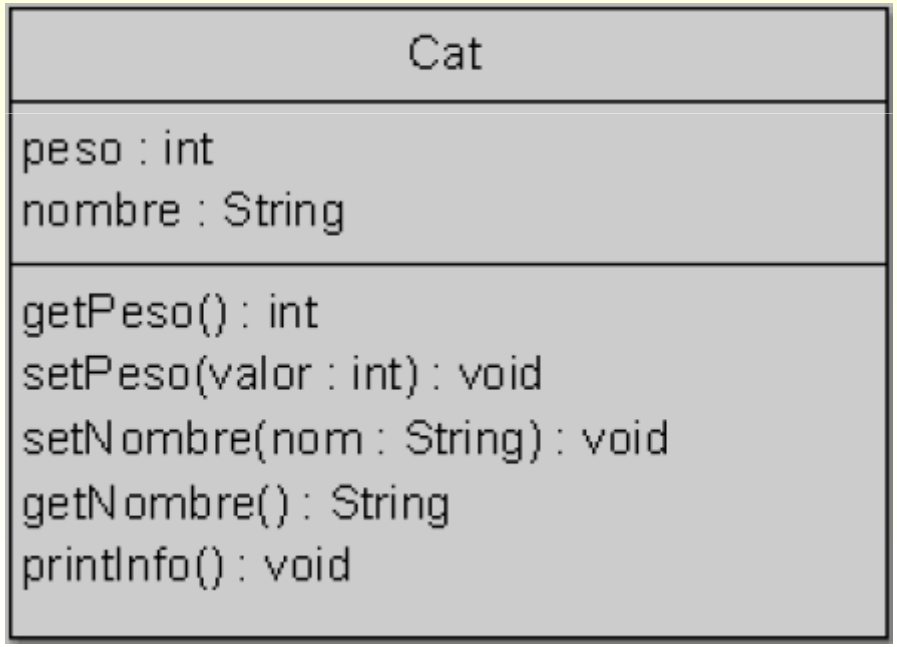
```
 // dando valores de inicialización
int[] a1 = {1, 2, 3, 4, 5};
int[] a2; // sólo declarando

a2 = a1; //copiando referencia
```

Ejemplo

- Generar la clase Arreglo que implemente dos métodos para:
  - Crear un arreglo con el alfabeto (26 letras). El método no recibirá datos pero devolverá el arreglo creado
  - Imprimir el contenido de un arreglo de caracteres recibido. El método no devolverá datos.

Ejemplo 2

- Implementar la clase indicada en el siguiente diagrama de clases:



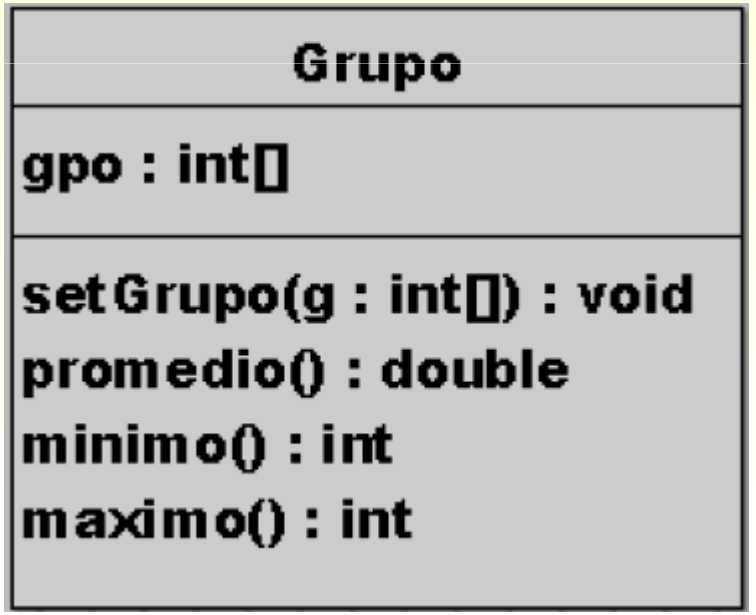
Solución clase PruebaCat

```
public class TestCat{
   public static void main (String[] args){
        Cat[] cs= new Cat[3];
        for(int t=0;t<cs.length;t++)
                cs[t] = new Cat();
        cs[0].setNombre("Felix");
        cs[0].setPeso(10);
        cs[1].setNombre("Garfield");
        cs[1].setPeso(30);
        // imprime el arreglo de objetos
        for(int k=0;k<cs.length;k++)
        {
                cs[k].printDatos();
        }
   } // fin main
} // fin clase
```

Ejercicio I

- Implementar una clase que proporcione métodos aplicables a un grupo de datos enteros. La clase **Grupo** implementa los siguientes métodos:

Implementar la clase **PruebaGrupo** que demuestre el uso de los métodos definidos



Ejercicio II

- Implemente en *main* lo siguiente:
  - Declare e inicialice un arreglo *a1* de enteros con valores aleatorios
  - Declare un arreglo *a2* y asigne a él el arreglo *a1*
  - Imprima los valores de ambos arreglos
  - Cambie el valor de un elemento de a1
  - Imprima los elementos de a2
  - ¿Qué sucede?

Copiando arreglos

- Cuando se copian arreglos, se copian referencias
- Para generar una copia independiente de los elementos se puede usar:
  - Ciclos

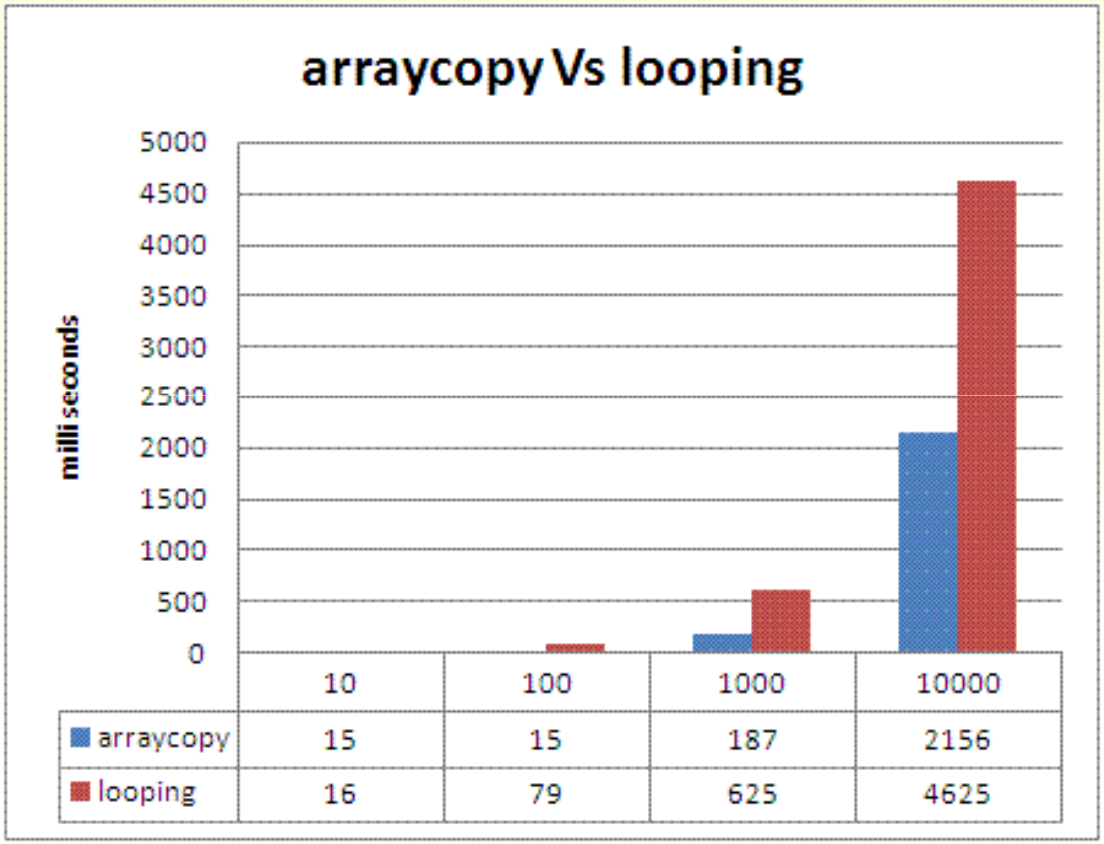
```
int[] a = {1, 2, 3, 4, 5};
int[] b = new int[5];

for(int i = 0; i < a.length; i++)
   b[i] = a[i];
```

Copiando arreglos II

- `System.arraycopy(src, srcPos, dest, destPos, length);`

```
char[] from = { 'd', 'e', 'c', 'a'};
char[] to   = new char[2];
System.arraycopy(from, 1, to, 0, 2);
```

Copiando arreglos



	10	100	1000	10000
arraycopy	15	15	187	2156
looping	16	79	625	4625

Elementos de prueba

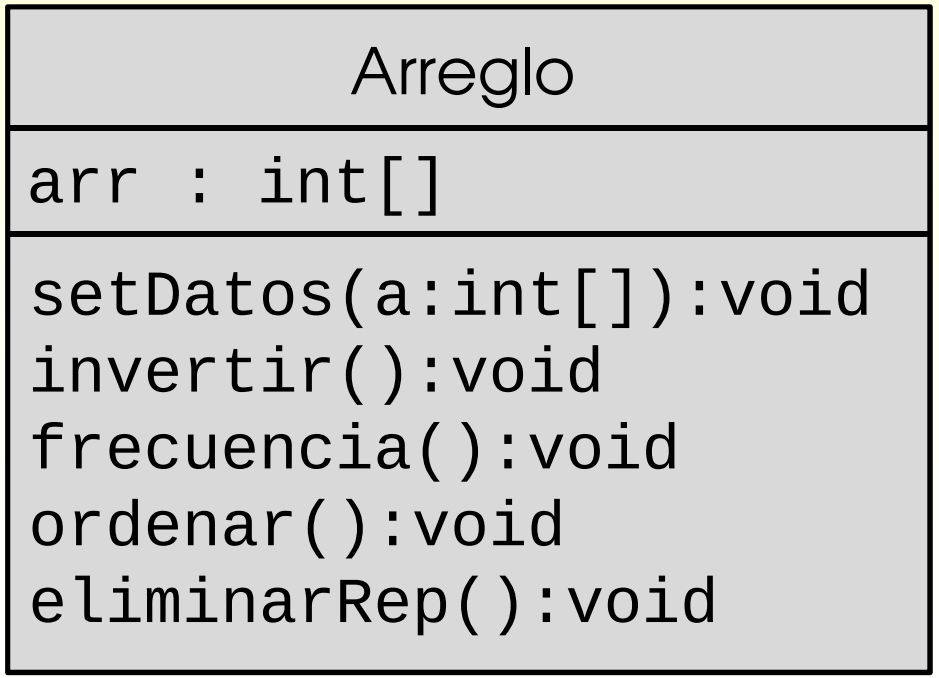
Milisegundos

Ejercicio II a

- Implemente en *main* lo siguiente:
  - Declare e inicialice un arreglo *a1* de enteros con valores aleatorios
  - Declare un arreglo *a2* y copie a él el arreglo *a1* usando el método *arraycopy*
  - Imprima los valores de ambos arreglos
  - Cambie el valor de un elemento de a1
  - Imprima los elementos de a2
  - ¿Qué sucede?

Ejercicio III

- Implemente la clase *Arreglo* bajo las siguientes consideraciones:
  - El ordenamiento será de menor a mayor.
  - El rango de valores que contengan el arreglo es de 1 a 99



Ejercicio III (continuación)

- Implemente la clase *TestArreglo* que demuestre el funcionamiento de la clase *Arreglo*, considerando que:
  - Los valores del arreglo se ingresarán desde consola:

```
c:\java Arreglo 25 12 33 99 72
```

Ejercicio IV

Sean A y B dos conjuntos de enteros

A={1,3,4,5}

B={4,3,5,6}

Conjunto
conjunto: int [ ]
setDatos(c:int[ ]): void interseccion(c:int[ ]):void union(c:int[]):void diferencia(c:int[]):void diferenciaSim(c:int[]):void

AUB = {1,3,4,5,6} → *union*

A∩B = {3,4,5} → *interseccion*

A-B = {1} → *diferencia*

A▲B = {1,6} = (A-B U B-A) =

(AUB) - (A∩B) → *diferenciaSim*

Ejercicio IV (continuación)

Ejemplo de implementación para la clase PruebaConjunto:

```
…
int[ ] a    = {1,3,4,5}
int[ ] b    = {1,6}
Conjunto c1 = new Conjunto();

c1.setDatos(a);
c1.union(b) // imprimirá la unión de a y b
c1.interseccion(b) // imprimirá intersección de a
                   // y b en pantalla

…
```