

Clase 15: Arreglos

En Java existe un tipo especial de objetos que permite almacenar gran cantidad de información: los arreglos.

1 Creación de un arreglo

La declaración de una variable que hará referencia a un objeto debe indicar el tipo de los elementos almacenados. La siguiente declaración indica que **p** hará referencia a un arreglo de enteros:

```
int p[];
```

La creación del arreglo, se realiza mediante `new`. Nuevamente la sintaxis es especial:

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

Entre los corchetes se indica la cantidad de elementos que se guardarán en el arreglo. Este número es fijo. Una vez que el arreglo es creado, su tamaño no puede modificarse (esta es una de las limitaciones importantes de los arreglos).

2 Acceso a los elementos

Cada elemento de un arreglo se puede acceder como si fuera un variable común y corriente. Basta con recordar que cada elemento tienen un índice (posición) dentro del arreglo, y que ese índice debe especificarse en cada acceso:

```
int var=p[5];
```

en este caso se le asigna el 6to elemento del arreglo **p** a la variable **var**.

3 Los índices

En un arreglo de `n` elementos, los índices van de 0 a `n-1`. Los índices siempre son números enteros. Escribir algo como `p[3.5]` no tiene sentido.

4 Problemas resueltos

4.1 Leer `n` números del teclado y escribirlos invertidos

```
1 con.println("Cuantos numeros?");
```

```
2 int n=con.readInt();
3 int a[]=new int[n];
4 for (int i=0; i<n; i++)
5     a[i]=con.readInt();
6 for (int i=n-1;i>=0;i--)
7     con.print(a[i]);
```

5 length

Cada objeto arreglo tiene una variable de instancia con la cantidad de elementos que contiene. Esto es particularmente útil cuando uno no conoce a priori el tamaño del arreglo, como por ejemplo, cuando un arreglo es pasado como parámetro de un método:

```
double suma(double a[]) {
    double s=0;
    for (int i=0; i<a.length; i++)
        s+=a[i];
    return s;
}
```

6 Métodos que devuelven arreglos

Cuando un método desea devolver más de un valor, se puede lograr usando arreglos. Por ejemplo, un método que recibe un nombre, como "Pedro Gonzalez" y devuelve el nombre y apellido, se puede escribir así

```
String[] separa(String nombreCompleto) {
    String p[]=new String[2];
    int i=nombreCompleto.indexOf(" ");
    p[0]=nombreCompleto.substring(0,i);
    p[1]=nombreCompleto.substring(i+1);
    return p;
}
```

7 Errores comunes

7.1 Copiar arreglos

Mucha gente copia un arreglo a otro de la siguiente forma:

```
int a[]=...
int b[];
b=a;
```

Al igual que en el caso de todos los objetos, este código solo hace que b haga referencia al mismo arreglo referenciado por a. La forma correcta de copiar un arreglo es

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

8 Limitaciones de los arreglos

1. No pueden modificar su tamaño
2. Eliminar un elemento de un arreglo de tamaño n, toma en promedio n/2 operaciones
3. Insertar un elemento en un arreglo ordenado, también toma en promedio n/2 operaciones

9 Arreglos de objetos

No hay que olvidar que los arreglos de objetos en realidad son arreglos de **referencias** a objetos. Por lo tanto, cuando uno crea un arreglo de objetos tiene que crear además cada objeto por separado:

```
Punto p[]=new Punto[5];
p[0]=new Punto(1,2);
p[1]=new Punto(4,0);
...
p[4]=new Punto(0,0);
```

10 Los arreglos en los métodos

La sintaxis para devolver arreglos es

```
int[] metodo() {...}
```

Para recibir arreglos, se debe especificar

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

11 Arreglos de dos dimensiones

Un arreglo puede tener más de una dimensión

```
int matriz[][]=new int[10][10];
for (int i=0; i<10; i++)
    for (int j=0; j<10; j++)
        matriz[i][j]=i*j;
```

Las dimensiones se pueden calcular como **matriz.length** y **matriz[0].length**