Estructuras de datos
Clase práctica 2

UNIVERSIDAD NACIONAL DE COLOMBIA
SEDE MEDELLÍN

Contenido

- Arreglos dinámicos

Material elaborado por: Julián Moreno

Facultad de Minas, Departamento de Ciencias de la Computación y la Decisión

Arreglos dinámicos

Ya vimos en clase teórica que la diferencia entre un arreglo “normal” y uno dinámico es que el segundo permite, luego de creado, agregar más elementos o eliminarlos. ¿Cómo podemos usarlos en Java?

Una forma de hacerlo es mediante la clase *Vector*, que ¿adivinen en que librería está?

```
Vector <Integer> miArreglo = new Vector<Integer>();
```

También puede ser Byte, Short, Long, Float, Double (comenzando con mayúscula)

Entre los métodos que tiene, los siguientes serán de nuestro interés:

Arreglos dinámicos

`add(e)`	Ingresa el elemento *e* al final del arreglo
`add(i,e)*`	Ingresa el elemento *e* en el índice *i* del arreglo
`clear()`	Borra todos los elementos
`get(i)*`	Devuelve el elemento en el índice *i*
`isEmpty()`	Devuelve verdadero si el arreglo está vacío
`indexOf(e)`	Devuelve el índice de la primera ocurrencia del elemento *e* dentro del arreglo, o -1 si no está
`lastIndexOf(e)`	Devuelve el índice de la última ocurrencia del elemento *e* dentro del arreglo, o -1 si no está
`remove(e)`	Borra la primera ocurrencia del elemento *e* si es que está, sino no hace nada. Atención: solo funciona para arreglos que no son de tipo `Integer`, en cuyo caso se recomienda usar `X.removeElementAt(X.indexOf(e))`
`removeElementAt(i)*`	Borra el elemento en el índice *i*
`set(i,e)*`	Reemplaza el elemento en el índice *i* por *e*
`size()`	Devuelve la cantidad de elementos en el arreglo

**i* debe ser un índice válido (*0 ≤ i ≤ n-1*)

Comparación

Operación	Arreglo estático	Arreglo dinámico
Declaración	`int a[ ] = new int[N]`	`Vector <Integer> a = new Vector<Integer>()` *
Indexado	`a[i]`	`a.get(i)`
Asignación / reemplazo	`a[i] = e`	`a.set(i, e)`
Búsqueda	`//Hay pue programarlo`	`a.indexOf(e)` `a.lastIndexOf(e)`

* Recuerden que también puede ser `Byte`, `Short`, `Long`, `Float`, `Double` (comenzando con mayúscula)

Ejemplo

```
import java.util.*;
public class Main {
    public static void main(String[] args) {
        int A[] = new int[3];
        Vector <Integer> B = new Vector <Integer>();
        A[0] = 10; A[1] = 20; A[2] = 30;
        B.add(40); B.add(60); B.add(1,50);
        System.out.println(B.indexOf(60));
        System.out.println(B.lastIndexOf(70));
        B.set(0, 45);
        System.out.println(B.get(0));
        B.removeElementAt(B.indexOf(50));
        System.out.println(B.size());
    }
}
```