Arreglos

Definición e implementación

Verónica E. Arriola-Rios

Facultad de Ciencias, UNAM

19 de septiembre de 2022

Definición

Temas

Tipo abstracto de datos: arreglo

Conjunto de datos:

Un *arreglo* es una estructura de datos que contiene un conjunto de **elementos** *del mismo tipo*, un conjunto de **índices** y un conjunto de operaciones que se utilizan para definir, manipular y abstraer estos elementos de datos.

Sengupta y Korobkin 2014

50	25	75	10	39	0	100	1	0	29	42	0	0	0	120
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Operaciones:

- Creación: dado su tamaño devuelve un arreglo.
- Asignación: dados los índices, guarda el valor en la posición correspondiente.
- Lectura: dados los índices, devuelve el valor almacenado en la posición indicada.

Arreglo multidimensional

Definición (Arreglo)

Un arreglo de

1. dimensión n,
2. de elementos de tipo X y
3. de tamaño $T = t_1 \times t_2 \times ... \times t_n$, donde t_j es el tamaño del arreglo en la j-ésima dimensión,

es un conjunto de T elementos de tipo X, en el que cada uno de ellos es identificado **unívocamente** por un **vector coordenado de n índices** $(i_1, i_2, ..., i_n)$, con $0 \leqslant i_j < t_j$.

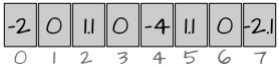
	0	1	2	3	4
0	50	25	75	10	39
1	0	100	1	0	29
2	42	0	0	0	120

Arreglos en Java

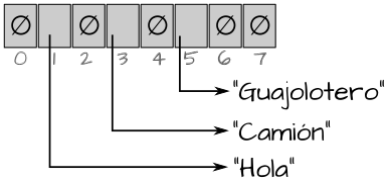
Arreglos en Java

En `Java` se distinguen dos tipos de arreglos:

- Arreglos de tipos primitivos: contienen al dato en su interior.



- Arreglos de objetos: contienen las direcciones de los objetos en el heap.



Temas

Arreglos de Primitivos

Se declaran e instancian con:

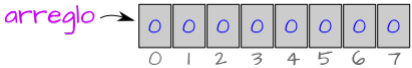
```
1  int[] arreglo;          // Vale null
2  int arreglo[];
3  arreglo = new int[8]; // Tamaño entre []
```

Ahora hay que asignar los valores:

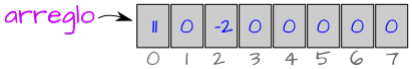
```
1  arreglo[0] = 11;
2  arreglo[2] = -2;
```

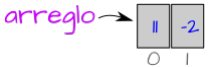
Creación rápida:

```
1  int arreglo[] = {11, -2};
```



Ojo: Sólo se reservan espacios para referencias a cadenas.





Arreglos de Objetos

Se declaran e instancian con:

```
1  String arreglo[];          // Vale null
2  arreglo = new String[8]; // Tamaño entre []
```

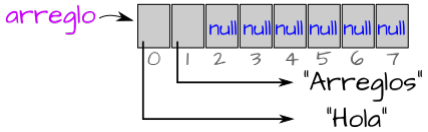
Ahora hay que crear los objetos y asignarlos:

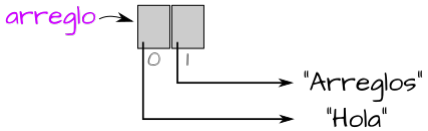
```
1  arreglo[0] = "Hola";
2  arreglo[1] = new String("arreglos");
```

Creación rápida:

```
1  String arreglo[] = {"Hola", "arreglos"};
```

Ojo: Sólo se reservan espacios para referencias a cadenas.





Ejemplo de uso

```java
package demos;

public class PrintCadenas{
  public static void main(String args[]){

    String cadenas[] = {"Hola", "Arreglos"};

    for(int i = 0; i < cadenas.length; i++){
        System.out.println(cadenas[i]);
    }
  }
}
```

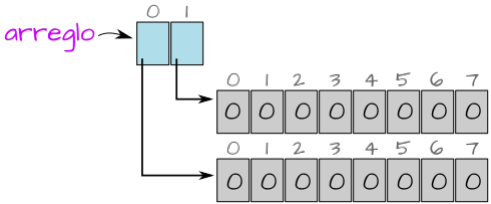
```
$ java demos.PrintCadenas
Hola
Arreglos
```

Temas

Arreglos 2D

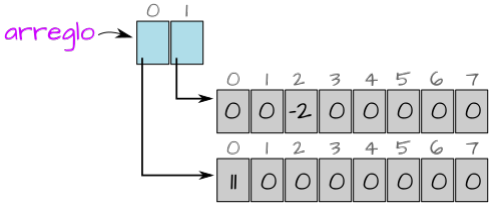
Se declaran e instancian con:

```
1  int[][] arreglo0;
2  int arreglo[][];
3  arreglo = new int[2][8];
```



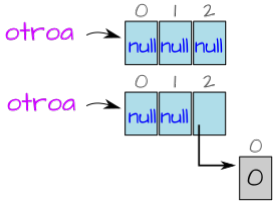
Ahora hay que asignar los valores:

```
1  arreglo[0][0] = 11;
2  arreglo[1][2] = -2;
```



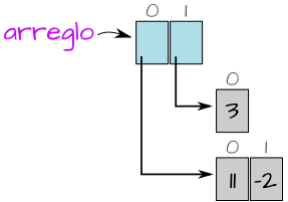
También se pueden crear renglón por renglón:

```
1  int[][] otroa = new int[3][];
2  otroa[2] = new int[1];
```



Creación rápida:

```
1  int arreglo[][] = {{11, -2}, {3}};
```



Excepciones

- Si se intenta asignar un valor a una posición que no existe, la ejecución del código se interrumpirá con una `ArrayIndexOutOfBoundsException`.

Código: ArrayIndexOutOfBoundsException

```
1 int[] a = new int[3];
2 a[0] = 3;                    // Correcto
3 a[3] = 8;                    // Incorrecto
4 System.out.println(a[4]);    // Incorrecto
```

- Si se intenta llamar una posición en un arreglo de más de 1D, y ese subarreglo no ha sido creado, obtendremos `NullPointerException`.

Código: NullPointerException

```
1 int[][] a = new int[3][];
2 a[0] = new int[2];
3 a[0][0] = 3;                 // Correcto
4 a[1][0] = 8;                 // Incorrecto
```

Arreglos nD

- El mecanismo de *arreglos de arreglos* que utiliza `Java` conocido como *Vectores de Iliffe*, permite crear fácilmente arreglos nD.
- En la sintaxis basta con agregar un par de `[]` por cada dimensión, tanto al declarar como al usar alguna posición.

Código: NullPointerException

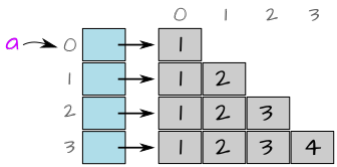
```
1 int[][][][] a = new int[3][8][3][2]; // Crear
2 a[2][7][0][1] = 7;                   // Asignar
3 int siete = a[2][7][0][1];           // Leer
```

Ejercicios interesantes

Temas

Base de la pirámide

```
1  public static int[][] base(int n) {
2    int[][] a = new int[n][];
3    for(int i = 0; i<n; i++) {
4      a[i] = new int[i+1];
5      for(int j = 0; j <= i; j++) {
6        a[i][j] = j+1;
7      }
8    }
9    return a;
10 }
```



Temas

Ordenamiento por inserción


```
1  public static void ordenaPorInsercion(double[] a) {
2    for(int i=1; i<a.length; i++) {
3      for(int j=i; j>1; j--) {
4        if(a[j] < a[j-1]) {
5          // Intercambia los valores
6          double temp = a[j];
7          a[j] = a[j-1];
8          a[j-1] = temp;
9        } else break;
10     }
11   }
12 }
```

Temas

Búsqueda binaria


```java
/** Busca el elemento en un arreglo ordenado.
 *  @returns el índice de la primer aparición o -1
 *           si no está. */
public static int busca(double[] a, double x) {
  int izq = 0, der = arr.length - 1;
  while (izq <= der) {
    int m = (izq + der) / 2;

    if (arr[m] == x) return m;    // Aquí está
    if (arr[m] > x) der = m - 1; // Busca a la izquierda
    else izq = m + 1;            // Busca a la derecha
  }
  return -1;  // No estuvo
}
```

Bibliografía

Bibliografía I

 Sengupta, Saumyendra y Carl Philip Korobkin (15 de feb. de 2014). *C++ Object-Oriented Data Structures*.

Licencia