Tema 10: Arreglos

Arrays

- Los arreglos son una colección de objetos del mismo tipo los cuales tienen un nombre en común y su acceso es mediante un índice.

- Declaración

  tipo dato[ ] identificador;

  tipo_dato identificador[];

- Creación

  identificador = new tipo_dato[tam];

- Cuando se crea un array sus elementos reciben el valor por default.

Declaración y creación

- La declaración y la creación pueden realizarse en la misma línea de código.

- Declaración
  int [ ] vec;

- Creación
  vec = new int[5];

- Declaración y creación juntas
  int [] vec = new int[5];

- Declaración e inicialización
  int [] vec = {10,20,30,40,50};

Inicialización

- Cuando se realiza una declaración e inicialización no es necesario la creación.

- Ejemplos:

```
int [] vec = {1*2,1*3,1*4,1*5};

String [] nombres ={“Ana”,”Julio”,”Inés” };

String nombre= “Manuel”;
String [] saludo = {“hola ”+nombre,
                    “adios ” + nombre };
```

Rango de los arrays

- Llamado también array bounds. Todos los arreglos son estáticos y su creación es dinámica sino se indica lo contrario.

- El índice en el cuál inician los elementos del arreglo es cero (0)

- Ejemplo:

```
String nombre= “Manuel”;
String [] saludo = new String[2];
saludo[0]=“hola ”+nombre;
saludo[1]=“adios ” + nombre;
```

- El número de elementos contenidos en el array es almacenado en el atributo implícito llamado **length**.

Ejemplo Arrays

- Reasignación de tamaño:

```
int [] myArray =new int[6];
myArray=new int[10];
```

```
public classArrays {
  public static void main(String[] args) {
    int[] a1 = { 1, 2, 3, 4, 5 };
    int[] a2;
    a2 = a1;

    for (int i = 0; i < a2.length; i++)
        a2[i]++;
    for (int i = 0; i < a1.length; i++)
         System.out.println( "a1[" + i + "] = " + a1[i]);
  }
}
```

Arrays multidimensionales

- Los arreglos de una dimensión son llamados vectores.
- Los arreglos de dos dimensiones son llamados matrices.
- De tres o más dimensione son llamados multidimensionales.
- Los arreglos de mas de una dimensión pueden ser creados de longitudes diferentes:

```
int [] dosDim= new int[4][];
dosDim[0]=new int[5];
dosDim[1]=new int[8];
```

Ejemplo matriz

- Ejemplo 1

```
int [][] matriz = new int [4][];
for (int i=0; i< matriz.length;i++){
    matriz[i] = new int [i+5];
    for (int j=0; j< matriz[i].length;j++){
          matriz[i][j] = i + j;
     }
 }
```

Ejemplo matriz

- Ejemplo 2

```
String[][] familias = {
 {"Picapiedra","Pedro","Wilma","Pebbles","Dino"},
 {"Mármol","Pablo","Betty","Bam Bam"},
 {"Simpson","Homer","Marge","Burt","Lisa","Maggie"}
};

for (int i = 0; i < familias.length; i++) {
   System.out.print(familias[i][0] + ": ");
   for (int j = 1; j < familias[i].length; j++) {
       System.out.print(familias[i][j] + " ");
   }
   System.out.println();
}
```

Ejemplo múltiple array 1a

```
import java.util.*;

public class MultiDimArray {
  static Random rand = new Random();
  static int pRand(int mod) {
    return Math.abs(rand.nextInt()) % mod + 1;
  }
  static void prt(String s) {
    System.out.println(s);
  }
  public static void main(String[] args) {
    int[][] a1 = { { 1, 2, 3, },
                   { 4, 5, 6, }  };
    for (int i = 0; i < a1.length; i++)
      for (int j = 0; j < a1[i].length; j++)
        prt("a1[" + i + "][" + j + "] = " + a1[i][j]);
```

Ejemplo múltiple array 1b

```
// Array de 3 dimensiones
    int[][][] a2 = new int[2][2][4];
    for (int i = 0; i < a2.length; i++)
       for (int j = 0; j < a2[i].length; j++)
          for (int k = 0; k < a2[i][j].length; k++)
             prt("a2[" + i + "][" + j + "][" + k + "] = " + a2[i][j][k]);

    // Array de 3 dimensiones de diferente longitud
    int[][][] a3 = new int[pRand(7)][][];
    for (int i = 0; i < a3.length; i++) {
        a3[i] = new int[pRand(5)][];
        for (int j = 0; j < a3[i].length; j++)
              a3[i][j] = new int[pRand(5)];
    }
    for (int i = 0; i < a3.length; i++)
       for (int j = 0; j < a3[i].length; j++)
         for (int k = 0; k < a3[i][j].length; k++)
          prt("a3[" + i + "][" +  j + "][" + k + "] = " + a3[i][j][k]);
```

Ejemplo múltiple array 1c

```
// Array de objetos (no primitivos)
    Integer[][] a4 = {  { new Integer(1), new Integer(2)},
                        { new Integer(3), new Integer(4)},
                        { new Integer(5), new Integer(6)} };
    for (int i = 0; i < a4.length; i++)
       for (int j = 0; j < a4[i].length; j++)
            prt("a4[" + i + "][" + j + "] = " + a4[i][j]);

    Integer[][] a5;
    a5 = new Integer[3][];
    for (int i = 0; i < a5.length; i++) {
        a5[i] = new Integer[3];
        for (int j = 0; j < a5[i].length; j++)
            a5[i][j] = new Integer(i*j);
    }
    for(int i = 0; i < a5.length; i++)
       for(int j = 0; j < a5[i].length; j++)
          prt("a5[" + i + "][" + j +  "] = " + a5[i][j]);
    }
}
```

Recorrido en arrays

- Los arreglos se pueden recorrer con un ciclo convencional usando el método length o bien con el for-each.

- Ejemplo

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

```
public void printElements (int [ ]array]){
    for (int element :array){
        System.out.println(element);
}
```

Arreglos de referencias

- Los arreglos pueden ser creados de un tipo de clase.

- Ejemplo

```
public Point[ ] createArray(){
    Point[ ] p;
    p =new Point[10];
    for (int i=0; i< 10;i++){
        p[i]= new Point(i,i+1);
   }
   return p;
}
```

Arreglos de referencias

- Inicialización

Ejemplo

```
public class MyDate{
   private int day;
   private int month;
   private int year;
   public MyDate(int day, int month,int year){
      this.day=day;
      this.month=month;
      this.year=year;
   }
}
// en el main
MyDate [ ] dates= {  new MyDate(7,7,1968);
                     new MyDate(22,8,1976);
                     new MyDate(5,7,2004);
                  };
```

Paso de parámetro

- Un arreglo se trata como un objeto, pasándose referencias entre variables.

- Cuando se llama a un método y se le pasa el arreglo, el método hace su copia de la referencia pero comparte el arreglo.

- Ejemplo

```
Void metodo(int [ ] x) { x[0]=2; }
int [ ] a=new int[5];
System... a[0] ;  // vale 0
metodo(a);
System..a[0];    // vale 2
}
```

Paso de parámetro

- Cuando una variable de tipo array se hace igual a otra, se copia la referencia pero se comparte el arreglo.

- Ejemplo

```
int [ ] a=new int[5];
System... a[0] ;  // vale 0
int [ ] b = a;
b[0] = 2;
System..a[0];    // vale 2
```

El array del main

- Encabezado:

```
public static void main( String args[] ){
  ...
}
```

- Esta línea especifica un método que el intérprete Java busca para ejecutar en primer lugar. Java utiliza la keyword *main* para especificar la primera función a ejecutar.

- **public** significa que el método main puede ser llamado por cualquiera, incluyendo el intérprete Java.

El array del main

- **static** indica al compilador que main se refiere a la propia clase **MiClase** y no a ninguna instancia de la clase.

  De esta manera, si alguien intenta hacer otra instancia de la clase, el método main no se instanciaría.

- **void** indica que main no devuelve nada.

- **args[]** es la declaración de un array de Strings. Estos son los argumentos escritos tras el nombre de la clase en la línea de comandos:

  $ java MiClase arg1 arg2 ...

Métodos para arrays

- Para copiar el contenido de un arreglo se pueden usar las dos formas siguientes:

- 1 Uso de clone

```
int [ ] a=....;
int [ ] b = (int []) a.clone();
```

Donde si los datos del arreglo son tipos primitivos se copia su valor y si son objetos se copia su referencia compartiéndose el objeto.

Métodos para arrays

- 2 Uso de java.lang.System.arraycopy(..)

- Sintaxis:

System.arraycopy(array_origen, indice1,array_destino,indice2,cantidad);

Donde: índice1 indica la posición del array1 donde se realizará la copia; indice2 indica la posición del array2 donde se almacenaran los datos copiados; cantidad indica el número de elementos copiados de array1 a array2.

- Ejemplo:

System.arraycopy(vector1,0,vector2,0,5);