



**UNIVERSIDAD NACIONAL DE
INGENIERÍA**



Lenguaje Orientado a Objetos

Nombre del docente(s):

Ing. Jacqueline del Rosario López Alvarado.
con la colaboración de: MSc. Ing. Janine Mairena Solórzano.

Fecha de elaboración:

07 de enero 2020

Semana



Objetivo de esta semana

Implementa nuevas estructuras de datos en el desarrollo de programas escritos en Java con el fin de guardar información en ambiente applet y aplicación.

1. ARREGLOS	4
• DEFINICIÓN	4
• ARREGLOS UNIDIMENSIONALES	4
• DECLARACIÓN DE ARREGLOS.....	4
• TAMAÑO DE ARREGLOS, ATRIBUTO LENGTH	6
• TAMAÑO DE ARREGLOS, ATRIBUTO LENGTH	6
• <i>Ejemplos con arreglos.....</i>	<i>6</i>
• <i>Métodos aplicables a los arreglos en Java</i>	<i>8</i>
• LONGITUD DE UN ARREGLO.....	8
• LA CLASE ARRAYS	9
• <i>Sintaxis alternativa para la declaración de arreglos en Java.....</i>	<i>10</i>
• <i>Ejemplos con arreglos – JApplets en Java.....</i>	<i>10</i>
• ARREGLOS BIDIMENSIONALES.....	16
• <i>Declaración de los arreglos bidimensionales.....</i>	<i>17</i>
2. BIBLIOGRAFÍA	19

1. Arreglos

Definición

Un *arreglo* es un grupo de variables del mismo tipo al que se hace referencia por medio de un nombre común. Se pueden crear arreglos de cualquier tipo, y pueden tener una dimensión igual a uno o mayor. Para acceder a un elemento concreto de un arreglo se utiliza su índice o subíndice, el cual es el número de la posición del elemento.

Los arreglos son *objetos*, por lo que se consideran como tipos de *referencia*. Lo que consideramos generalmente como un arreglo es en realidad una referencia a un objeto arreglo en memoria. Los elementos de un arreglo pueden ser tipos primitivos o de referencia. Para hacer referencia a un elemento específico en un arreglo, debemos especificar el nombre de la referencia al arreglo y el número de la posición del elemento en el arreglo.

Arreglos unidimensionales

Un *arreglo unidimensional* es, esencialmente, una lista de variables del mismo tipo.

El arreglo se crea a través de una variable del tipo deseado. La forma general de declarar un arreglo unidimensional es:

Declaración de Arreglos

Se declara de modo similar a otros tipos de datos, excepto que se debe indicar al compilador que es un arreglo y esto se hace con corchetes.

```
int [ ] v;  
float w[ ];
```

Los corchetes se pueden colocar de dos formas:

- A continuación del tipo de datos
- Después del nombre del arreglo

La sintaxis de declaración de variables arreglo en Java es:

```
Tipo nombre [ ];
```

En donde, *tipo* declara el tipo base del arreglo, el cual determina el tipo de cada elemento que conforma el arreglo.

Ejemplo:

```
int días del mes;
```

Aunque esta declaración establece que **días_del_mes** es una variable de tipo arreglo, todavía no existe realmente ningún arreglo. De hecho, el valor de **días_del_mes** es **null**. Es importante indicar que la palabra reservada **null** representa un arreglo que no tiene ningún valor. Para que **días_del_mes** sea un verdadero arreglo de enteros se debe reservar espacio utilizando el operador *new* y asignar este espacio a **días_del_mes**. *new* es un operador especial que reserva espacio de memoria. La forma general del operador **new** cuando se aplica a arreglos unidimensionales es la siguiente:

```
identificador = new tipo[tamaño];
```

donde *tipo* especifica el tipo de datos almacenados en el arreglo, *tamaño* especifica el número de elementos, *identificador* es la variable a la que se asigna el nuevo arreglo. Tras la declaración del array, se tiene que iniciar. Eso lo realiza el operador **new**, que es el que realmente crea el array indicando un tamaño. Cuando se usa **new** es cuando se reserva el espacio necesario en memoria y se inicializan a cero automáticamente todos sus elementos. Un array no inicializado es un array **null**.

El siguiente ejemplo reserva espacio para un arreglo de 12 elementos enteros y los asigna a **días_del_mes**.

```
días_del_mes = new int[12];
```

Cuando se ejecute esta sentencia, **días_del_mes** hará referencia a un arreglo de 12 elementos enteros inicializándolos todos en cero y algunas veces se le denomina **elemento cero**.

Resumiendo, la obtención de un arreglo es un proceso que consta de dos partes:

1. declarar una variable del tipo de arreglo deseado.
2. reservar espacio de memoria para almacenar el arreglo mediante el operador **new**, y asignarlo a la variable.

En Java, la memoria necesaria para arreglo se reservan dinámicamente. Una vez reservada la memoria para un arreglo, se puede acceder a un elemento concreto del arreglo especificando su índice dentro de corchetes. Todos los índices de un arreglo comienzan en cero.

🔗 **Tamaño de Arreglos, atributo length**

Java considera cada arreglo como un objeto que, además de tener capacidad para almacenar elementos, dispone del atributo `length` con el número de éstos.

```
double [ ] v = new double[15];  
System.out.print(v.length); //escribe 15, número de elementos de v.
```

Java conoce el número de elementos de un arreglo cuando se crea con el operador **new** o con una expresión de inicialización; **length** está protegido y no puede ser modificado, ya que se define con el calificador final.

Ejemplo:

```
double suma (double [ ] w)  
{  
    double s = 0.0;  
    for (int i = 0; i < w.length; i++)  
        s += w[i];  
    return s;  
}
```

🔗 **Tamaño de Arreglos, atributo length**

Java considera cada arreglo como un objeto que, además de tener capacidad para almacenar elementos, dispone del atributo `length` con el número de éstos.

🚦 **Ejemplos con arreglos**

#1. Programa en Java que guarda datos de tipo entero en un arreglo de tres elementos y e imprime la suma total de este.

```
package arreglos;  
import javax.swing.JOptionPane;  
public class Arreglos {  
  
    public static void main(String[] args) {  
        int notas[ ],N1,suma=0;  
        String n1;  
        notas=new int[3]; //indica que el arreglo tendrá 3 elementos  
        //ahora se solicita los datos al usuario para estas variables
```

```

for(int i=0;i<3;i++)
{
    n1=JOptionPane.showInputDialog("Ingrese el elemento del arreglo: ");
    N1=Integer.parseInt(n1);//convertir los numeros de tipo String a int
    notas[i]=N1; //guardamos en el arreglo
    suma+=notas[i];
} //fin del for
JOptionPane.showMessageDialog(null, "La suma de los numeros guardados en el arreglo
es: "+suma,"Resultados",JOptionPane.PLAIN_MESSAGE);
}

}

```

#2. Programa en Java que inicializa explícitamente un arreglo de tipo entero y evalúa quien es el mayor de estos imprimiéndolo en pantalla.

```

package arreglos;
import javax.swing.JOptionPane;
public class Arreglo1 {
    public static void main(String[] args) {
        int n[]=new int[ ] {1,2,3,4,5,6,7,8,9}; //declaración del arreglo e inicializacion del mismo
        int mayor=n[0]; //se asigna el primer elemento del arreglo a la variable mayor

        //el ciclo permitirá evaluar cada elemento del arreglo y evaluar quien es el mayor
        for(int i=0;i<9;i++)
        {
            if(n[i]>mayor); //guardamos en el arreglo
            mayor=n[i]; //actualizo el valor de mayor
        } //fin del for
        JOptionPane.showMessageDialog(null, "El número mayor de la lista es: "+mayor
        ,"Número mayor",JOptionPane.PLAIN_MESSAGE);
    }

}

```

#3. Programa en Java que solicita las dimensiones de un arreglo bidimensional y luego llena el arreglo.

```

//cadenas de caracteres
import javax.swing.*;
public class cadenas {

```

```

public static void main(String[] args) {
    int i,j,fil,col;
    String cadena="";
    fil=Integer.parseInt(JOptionPane.showInputDialog("Ingresa el numero de filas de la
matriz: "));
    col=Integer.parseInt(JOptionPane.showInputDialog("Ingresa el numero de columnas
de la matriz: "));
    int[][]matriz=new int[fil][col];
    for(i=0;i<fil;i++){
        for(j=0;j<col;j++){
            matriz[i][j]=Integer.parseInt(JOptionPane.showInputDialog("Ingresa los valores
"+i+"["+j+"]"));
            cadena+=matriz[i][j]+"\\t";

        }
    }
    JOptionPane.showMessageDialog(null,"Los valores ingresados son: ");
    for(i=0;i<fil;i++){
        for(j=0;j<col;j++){
            JOptionPane.showMessageDialog(null,matriz[i][j]);

        }
    }
    JOptionPane.showMessageDialog(null,"Los valores ingresados en la matriz son:
"+cadena);
}
}

```

Métodos aplicables a los arreglos en Java

Longitud de un arreglo

```

System.out.println(notas.length); //Sale 5
System.out.println(notas[2].length); //Sale 400

```

Ejemplo 3:

Programa en Java que inicializa e imprime la longitud de un arreglo inicializado explícitamente utilizando el método **length**.


```
package arreglos;
public class Arreglo2 {

    public static void main(String[] args) {
        int n[]=new int[ ] {1,2,3,4,5,6,7,8,9}; //declaración del arreglo e inicializacion del mismo
        System.out.println("longitud del arreglo: "+n.length);
    }

}
```

📌 La clase Arrays

En el paquete **java.util** se encuentra una clase estática llamada **Arrays**. Una clase estática permite ser utilizada como si fuera un objeto (como ocurre con **Math**). Esta clase posee métodos muy interesantes para utilizar sobre arrays.

```
Arrays.método(argumentos);
```

fill

Permite rellenar todo un array unidimensional con un determinado valor. Sus argumentos son el array a rellenar y el valor deseado:

```
int valores[]=new int[23];
Arrays.fill(valores,-1);//Todo el array vale -1
```

También permite decidir desde qué índice hasta qué índice rellenamos:

```
Arrays.fill(valores,5,8,-1);//Del elemento 5 al 7 valdrán -1
```

equals

Compara dos arrays y devuelve **true** si son iguales. Se consideran iguales si son del mismo tipo, tamaño y contienen los mismos valores.

sort

Permite ordenar un array en orden ascendente. Se pueden ordenar sólo una serie de elementos desde un determinado punto hasta un determinado punto.

```
int x[]={4,5,2,3,7,8,2,3,9,5};
Arrays.sort(x);//Estará ordenado
Arrays.sort(x,2,5);//Ordena del 2º al 4º elemento
```

binarySearch

Permite buscar un elemento de forma ultrarrápida en un array ordenado (en un array desordenado sus resultados son impredecibles). Devuelve el índice en el que está colocado el elemento.

Ejemplo:

```
int x[]={1,2,3,4,5,6,7,8,9,10,11,12};  
Arrays.sort(x);  
System.out.println(Arrays.binarySearch(x,8)); //Da 7
```

Sintaxis alternativa para la declaración de arreglos en Java

Existe una segunda forma de declarar un arreglo:

tipo[] nombre;

Aquí los corchetes siguen al especificador de tipo, y no al nombre del arreglo. Por ejemplo, las dos declaraciones siguientes son equivalentes:

```
int a1[ ] = new int[3];  
int[ ] a2 = new int[3];
```

Las siguientes declaraciones también son equivalentes:

```
char dosd1[ ] [ ] = new char[3] [4];  
char[ ] [ ] dosd2 = new char[3] [4];
```

Esta forma alternativa de declaración resulta conveniente cuando se declaran varios arreglos al mismo tiempo.

Ejemplo:

```
int [ ] nums, nums2, nums3; // crea tres arreglos
```

Ejemplos con arreglos – JApplets en Java

- Crear un arreglo de 10 elementos y solicitar al usuario el ingreso de estos valores. Se imprime en pantalla a través de JTextArea.

Nota: JTextArea

Uno de los componentes de la GUI basado en texto es ***JTextArea***, el cual permite que se muestren o se introduzcan varias líneas de texto. ***JTextArea*** permite un área para manipular varias líneas de texto.

```
//Creando un arreglo
package arreglos;

import javax.swing.*;

public class Arreglo3 {

    public static void main(String[] args) {
        int array[]; //se declara la referencia del arreglo.
        array=new int[10]; //se crea el arreglo
        String arreglo[]=new String[10]; //se guarda el número como cadena
        String salida="Índice\tValor\n";
        String ne=JOptionPane.showInputDialog("Digite el número de elementos: ");
        int n=Integer.parseInt(ne);
        for(int i=0;i<n;i++)
        {
            arreglo[i]=JOptionPane.showInputDialog("Digite el elemento: "); //se solicita el dato
            como una cadena.
            array[i]=Integer.parseInt(arreglo[i]); //se guarda en array[i] como entero.
        } //fin del for
        for(int cont =0;cont<arreglo.length;cont++) //el método length sobre arreglo deduce el
        tamaño del arreglo.
            salida+=cont+"\t"+array[cont]+" \n";
        JTextArea areaSalida=new JTextArea();
        areaSalida.setText(salida);
        JOptionPane.showMessageDialog(null,areaSalida,"Inicialización de un arreglo de valores
        int", JOptionPane.INFORMATION_MESSAGE);
        System.exit(0);

    }

} //end main
} //end class arreglo
```

Inicialización de un Arreglo con una decl... X



Indice	Valor
0	32
1	27
2	64
3	18
4	95
5	14
6	90
7	70
8	60
9	37

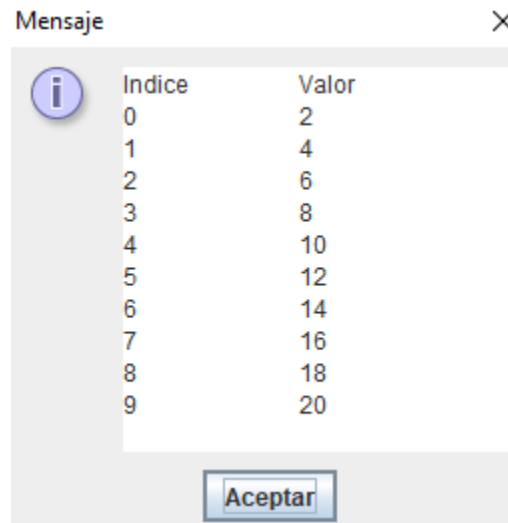
Aceptar

- b) Crear un arreglo de 10 elementos y solicitar al usuario el ingreso de estos valores. Se imprime en pantalla a través de JTextArea.

```
package arreglos;
import javax.swing.*;

public class Arreglo5 {
    public static void main(String[] args) {
        final int longi=10; //constante
        int array[];
        array=new int[longi];

        for(int cont=0;cont<array.length;cont++)
            array[cont]=2+2*cont;
        String salida="Indice\tValor\n";
        for(int cont=0;cont<array.length;cont++)
            salida+=cont+"\t" + array[cont]+"\\n";
        JTextArea areaSalida=new JTextArea();
        areaSalida.setText(salida);
        JOptionPane.showMessageDialog(null,areaSalida);
        System.exit(0);
    }
}
```



c) Programa que construye un Histograma desde Java

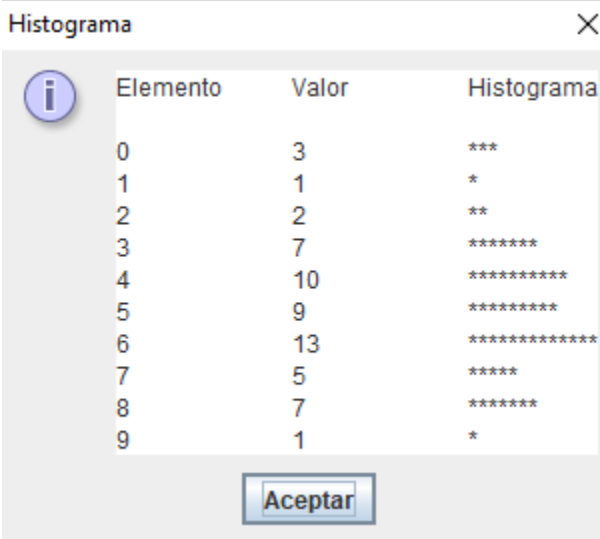
```
package arreglos;
import javax.swing.*;
public class Arreglo6 {

    public static void main(String[] args) {
        int array[] = {3,1,2,7,10,9,13,5,7,1};

        String salida="Elemento\tValor\tHistograma\n";

        for(int cont=0;cont<array.length;cont++)
        {
            salida+="\n" + cont+"\t" + array[cont]+"\t";
            //imprime barra de asteriscos
            for(int ast=0;ast<array[cont];ast++)
                salida+="*";
            //fin del for
            JTextArea areaSalida=new JTextArea();
            areaSalida.setText(salida);

            JOptionPane.showMessageDialog(null,areaSalida,"Histograma",JOptionPane.INFORMATION_MESSAGE);
            System.exit(0);
        }
    }
}
```



Elemento	Valor	Histograma
0	3	***
1	1	*
2	2	**
3	7	*****
4	10	*****
5	9	*****
6	13	*****
7	5	*****
8	7	*****
9	1	*

- d) La cafetería "Delicias María" en su necesidad de dar un servicio de calidad a sus clientes, realizará encuesta en donde se pidió a 40 estudiantes que calificarán la cafetería, en una escala 1-10(en donde 1 significa pésimo y 10 excelente). Guarde las 40 respuestas en un arreglo y sintetice los resultados de la encuesta.

```
public class Encuesta {
    public static void main(String[] args) {
        int respuestas [ ] = {1,2,6,4,8,5,9,7,8,10,1,6,3,8,6,10,3,8,2,7,6,5,7,6,8,6,7,5,6,6,5,6,7,5,6,4,8,6,8,10};
        int frecuencia[]=new int [11];
        //Para cada respuesta, seleccionar el elemento de respuesta y usar ese valor con índice en frecuencia para
        determinar el elemento a incrementar
        for (int resp=0; resp<respuestas.length;resp++)
            ++frecuencia[respuestas[resp]];
        String salida="Calificación\tFrecuencia\n";
        //Anexar la frecuencia al String de salida
        for(int calif=1;calif<frecuencia.length;calif++)
            salida+=calif+"\t"+frecuencia[calif]+"\\n";

        JTextArea areaSalida=new JTextArea();
        areaSalida.setText(salida);
        JOptionPane.showMessageDialog(null,areaSalida,"Programa de votación de
        estudiantes",JOptionPane.INFORMATION_MESSAGE);

        } //Fin del main
    }Fin de la clase
```

- e) Escribir una aplicación que lea una palabra y dentro de un método que imprima la palabra de la siguiente manera: ejemplo

```
import javax.swing.*;

public class Palabra {
    public static void main(String[] args) {
        String ppalabra=JOptionPane.showInputDialog("Introduzca palabra");
        triangulo(ppalabra);
    }

    //declaracion del metodo triángulo
    public static void triangulo(String ppalabra)
    {
        String imp="";

        for(int i=ppalabra.length() ; i>0 ; i--)
            imp +=ppalabra.substring(0,i)+"\\n";
        JOptionPane.showMessageDialog(null,imp);
    }
}
```

```

    }
}

```

Arreglos bidimensionales

Son conocidos como tablas o matrices y se representan mediante arrays bidimensionales. En Java se declara como un array de objetos array.

```
tipo nombre [ ] [3] ;
```

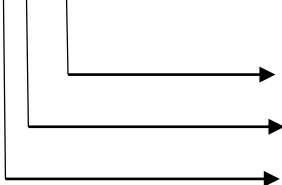
Las tablas de valores constan de información dispuesta en filas y columnas. Para identificar un elemento en concreto de una tabla se deben especificar los dos subíndices (*por convenio, el primero identifica la **fila** del elemento y el segundo identifica la **columna** del elemento*).

	0	1	2	3
0				
1				
2				

Tabla 1. Arreglo bidimensional

En la *Tabla 1.* se ilustra un array de doble subíndice, al mismo le denominaremos **a**, con tres filas y cuatro columnas (o sea un array 3x4). Un array con rn filas y n columnas se denomina array m_por_n.

	Columna 0	Columna 1	Columna 2	Columna 3
Fila 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Fila 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Fila 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]



Subíndice columna

Subíndice fila

Nombre del array

Cada elemento del array **a** se identifica como tal elemento, mediante el formato **a[i][j]** ; **a** es el nombre del array, e i, j son los subíndices que identifican unívocamente la fila y la columna de cada elemento de **a**.

Todos los elementos de la primera fila comienzan por un primer subíndice 0 y los de la columna cuarta tienen un segundo subíndice de 3 (4- 1).

Declaración de los arreglos bidimensionales

Los arrays bidimensionales se declaran con un par de corchetes para cada dimensión del array. Por ejemplo:

tipo nombre [][] ;



Declaración:

```
int [ ][ ]tablero;
```

Construcción:

```
tablero=new int[8][8];
```

Inicialización explícita:

```
int [ ][ ]tablero={ {1,3,5},{3,6,7} };
```

- ❖ Programa en Java para crear e inicializar un arreglo de 8 filas y 10 columnas con 80 enteros todos con el valor 3. Imprima en pantalla esos valores.

```

package arreglos;

public class bidimensional {

    public static void main(String[] args) {
        int numero[][];
        numero=new int [8][10];
        for(int x=3;x<numero.length;++x)
            for(int y=3;y<numero[x].length;++y){
                numero[x][y]=3;
                System.out.print(" "+numero[x][y]);
            }
    }
}

```

- ❖ Programa en Java para crear e inicializar un arreglo de 8 filas y 10 columnas con 80 enteros todos con el valor 3. Imprima en pantalla esos valores.

```

//programa que
package arreglos;

public class tabla1 {

    private static int c;

    public static void main(String[] args) {
        int t[][] , l=2;
        t=new int [2][4];
        for(int f=0;f<2;f++){
            for(int c=0;c<4;c++){
                t[f][c]=1;
                System.out.print(" "+t[f][c]);
                l+=2;
            } //fin del for externo

            System.out.println();

        } //fin del for interno

    }

}

```

2. Bibliografía

Aguilar, L. J. (1996). *Programación Orientada a Objetos Conceptos, Modelado, Diseño y Codificación en C++*. Madrid: McGraw-Hill.

Graham, I. (1996). *Métodos Orientados a Objetos*. Addison - Wesley/Díaz de Santos.

Herbert, S. (2009). *Manual de referencia Java*. Mc-Graw-Hill Internacional Editores, S.A de C.V.