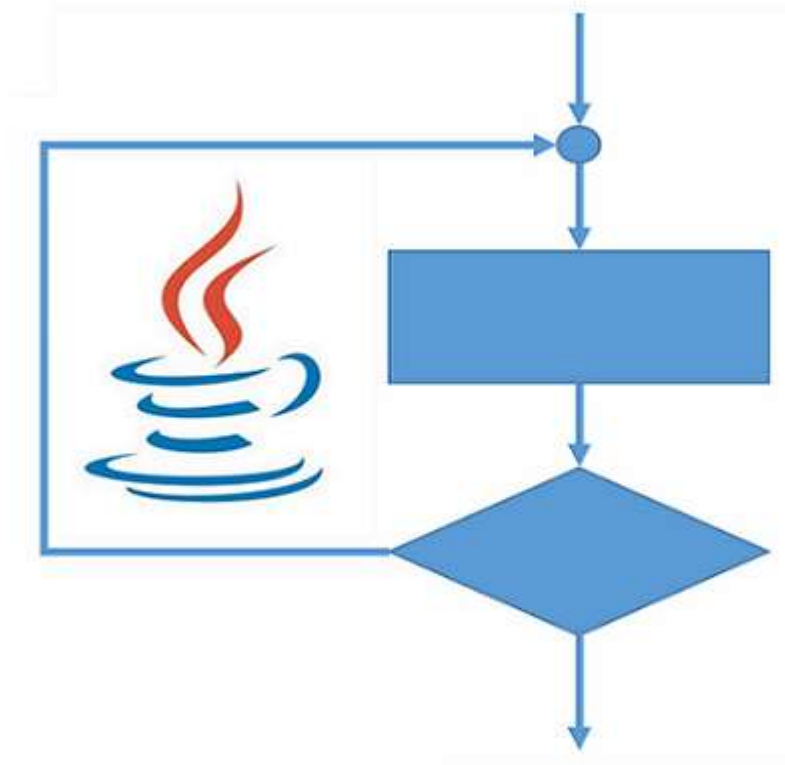


FUNDAMENTOS DE PROGRAMACIÓN CON JAVA



UNIDAD 07 ESTRUCTURAS DE DATOS

Eric Gustavo Coronel Castillo
youtube.com/DesarrollaSoftware
gcoronel@uni.edu.pe

INDICE

ARREGLOS	3
¿QUÉ ES UN ARREGLO?	3
ARREGLOS UNIDIMENSIONALES – VECTORES	4
DECLARACIÓN DE ARREGLOS	4
CREACIÓN DE ARREGLOS	5
ACCESO A ELEMENTOS DE UN ARREGLO	5
INICIALIZACIÓN UN ARREGLO	6
AVERIGUAR EL TAMAÑO DE UN ARREGLO	6
APLICACIÓN DE ARREGLOS UNIDIMENSIONALES	7
GENERACIÓN DE NÚMEROS ALEATORIOS	7
EJEMPLO 1.....	7
EJEMPLO 2.....	9
EJEMPLO 3.....	11
EJEMPLO 4.....	13
ARREGLOS BIDIMENSIONALES – MATRICES	15
DECLARACIÓN DE ARREGLOS BIDIMENSIONALES.....	16
CREACIÓN DE ARREGLOS	17
ACCESO A LOS ELEMENTOS DE UNA MATRIZ.....	18
INICIALIZACIÓN DE UNA MATRIZ	19
APLICACIÓN DE ARREGLOS BIDIMENSIONALES	20
EJEMPLO 5.....	20
EJEMPLO 6.....	22
CURSOS RELACIONADOS	24

ARREGLOS

El uso de variables es la forma más simple de guardar datos en memoria durante la ejecución de un programa, pero resulta inadecuado para algunos procesos, por eso debemos recurrir a una estructura de datos que permita almacenar varios datos como una sola unidad, para luego procesarlos mediante bucles, y una de las posibilidades son los arreglos.

¿QUÉ ES UN ARREGLO?

Un arreglo (array en inglés) es una estructura de datos conformada por un conjunto de variables del mismo tipo, agrupadas bajo un mismo nombre, a las cuales accedemos mediante un índice.



Figura 1 Ejemplo de un arreglo.

En la Figura 1 tienes un ejemplo de lo que sería un arreglo de tipo **String**, todos sus elementos son cadenas, en este caso los nombres de mis amigos.

El primer elemento tiene índice cero (0), y el último es el tamaño del arreglo disminuido en uno. Si tenemos un arreglo de tamaño 6, el primer elemento tendrá índice 0 y el último tendrá índice 5.

Los arreglos pueden ser de una dimensión, se les denomina vectores; de dos dimensiones, se les denomina matrices; podemos construir de más dimensiones, pero no es usual.

ARREGLOS UNIDIMENSIONALES – VECTORES

Los arreglos unidimensionales o vectores, representan una lista de variables contiguas y homogéneas (del mismo tipo) a las cuales podemos acceder mediante un índice, tal como se representa en la Figura 2.

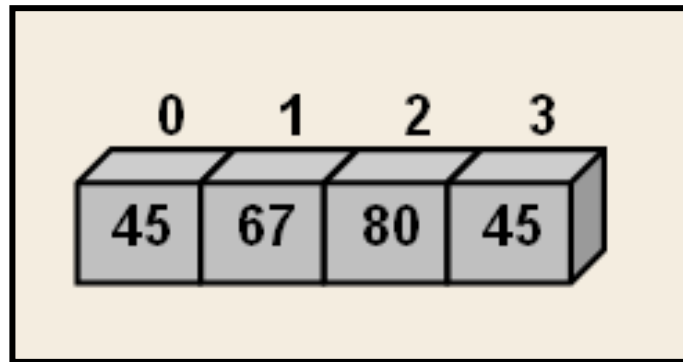


Figura 2 Arreglo Unidimensional

Declaración de Arreglos

Sintaxis

```
tipo variable_arreglo[];
```

o

```
tipo[] variable_arreglo;
```

En este caso solo estamos creando la variable que va a contener el arreglo, más no el arreglo en sí.

A continuación, tienes algunos ejemplos ilustrativos:

```
int lista1[];  
char lista2[];  
String lista3[];
```

Creación de Arreglos

Para la creación de los arreglos debemos utilizar el operador new.

Sintaxis

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

Es necesario haber declarado la variable **variable_arreglo** previamente.

A continuación, tienes algunos ejemplos ilustrativos:

```
lista1 = new int[10];      // De tamaño 10 (10 elementos)
lista2 = new char[15];    // De tamaño 15 (15 elementos)
lista3 = new String[8];   // De tamaño 8 (8 elementos)
```

Acceso a Elementos de un Arreglo

Para acceder a los elementos de un arreglo debes usar el índice del elemento.

Sintaxis

```
variable_arreglo[índice]
```

Por ejemplo, para guardar en la posición 3 del arreglo **lista3** el nombre **Claudia**, la instrucción es:

```
lista[3] = "Claudia";
```

Para imprimir el contenido de lista1, las instrucciones son:

```
for( int k = 0; k < 10; k++ )
    System.out.println( lista1[k] );
```

Inicialización un Arreglo

Puedes crear e inicializar un arreglo al mismo tiempo.

Sintaxis

```
tipo variable_arreglo[] = { elemento1, elemento2, . . . } ;
```

A continuación, tienes un ejemplo ilustrativo:

```
String ciudades[] = { "Trujillo", "Chiclayo", "Piura", "Tumbes" } ;

for( String dato: ciudades )
    System.out.println( dato );
```

Averiguar el Tamaño de un Arreglo

Los arreglos tienen una serie de propiedades y métodos entre los que encuentra la propiedad **length**, esta propiedad te permite obtener la longitud del arreglo.

La propiedad **length** te puede ayudar a realizar el recorrido del arreglo si utilizas un bucle, como se ilustra en el siguiente ejemplo:

```
String dbs[] = { "MySQL", "PostgreSQL", "Oracle", "SQL Server" } ;

for( int k = 0; k < dbs.length; k++ )
    System.out.println( dbs[k] );
```

APLICACIÓN DE ARREGLOS UNIDIMENSIONALES

Generación de Números Aleatorios

En java tienes el método **random()** que pertenece a la clase **Math**. Este método genera números aleatorios mayores o iguales que cero (0) y menores que uno (1).

Ejemplo 1

En este ejemplo se muestra como ordenar un vector de números enteros generados en forma aleatoria en el rango de 1 a 20.

En el arreglo **lista** se genera la lista de números aleatorios, el tamaño de la lista se pasa como parámetro en la línea de comandos, luego se saca una copia en **lista2**, El arreglo **lista2** es el que se ordena aplicando el método burbuja, y finalmente se imprimen ambas listas.

```
public class prog0701
{
    public static void main(String[] args)
    {
        int n = Integer.parseInt( args[0] );
        int lista[] = new int[n]; // Lista original
        int lista2[]; // Lista ordenada

        // Generar Lista
        for( int k = 0; k < n; k++ )
            lista[k] = (int)( Math.random() * 20 + 1 );

        // Sacar una copia de lista en lista2
        lista2 = lista.clone();

        // Ordenar Lista - Metodo Burbuja
        for( int i = 0; i < (n-1); i++ ) {
            for( int j = i + 1; j < n; j++ ) {
                if( lista2[i] < lista2[j] ) {
                    int temp = lista2[i];
                    lista2[i] = lista2[j];
                    lista2[j] = temp;
                }
            }
        }
    }
}
```

```
// Imprimir lista ordenada
System.out.println( "Lista Generada\tLista Ordenada" );
for( int k = 0; k < n; k++ ) {
    System.out.println( "\t" + lista[k] + "\t\t" + lista2[k] );
}

System.out.println( "\t----- Fin -----" );

}
}
```


Ejemplo 2

El siguiente ejemplo genera una lista de **n** números aleatorios, luego los ordena, para finalmente encontrar la Mediana.

Para encontrar la mediana primero se debe ordenar la lista, si el número de elementos es impar, la mediana es el número central de la lista, pero si el número de elementos es par, la mediana es el promedio de los dos números centrales.

```
public class prog0702
{
    public static void main(String[] args)
    {
        int n = Integer.parseInt( args[0] );
        int lista[] = new int[n]; // Lista original
        int lista2[]; // Lista ordenada

        // Generar Lista
        for( int k = 0; k < n; k++ ) {
            lista[k] = (int)( Math.random() * 100 + 1 );
        }

        // Ordenar Lista - Metodo Burbuja
        for( int i = 0; i < (n-1); i++ ) {
            for( int j = i + 1; j < n; j++ ) {
                if( lista[i] < lista[j] ) {
                    int temp = lista[i];
                    lista[i] = lista[j];
                    lista[j] = temp;
                }
            }
        }

        // Obtener la Mediana
        int p = n / 2;
        float mediana;
        if( n % 2 == 0 )
            mediana = (float)( lista[p] + lista[p-1] ) / 2;
        else
            mediana = lista[p];

        // Imprimir lista ordenada
```

```
System.out.println( "Lista Generada" );  
for( int k = 0; k < n; k++ )  
    System.out.println( "\t" + lista[k] );  
  
// Imprimir mediana  
System.out.println( "Mediana: " + mediana );  
System.out.println( "----- Fin -----" );  
  
}  
}
```

Ejemplo 3

En este ejemplo tenemos una lista de amigos, de lo que se trata es de ubicar la posición de uno de ellos, para lo cual ingresamos el nombre del amigo a buscar en la línea de comando como parámetro.

```
public class prog0703
{
    public static void main(String[] args)
    {
        int N = 10;
        String amigos[] = new String[N];
        String nombre = args[0];

        // Lista de amigos
        amigos[0] = "Sergio";
        amigos[1] = "Claudia";
        amigos[2] = "Hugo";
        amigos[3] = "Delia";
        amigos[4] = "Julio";
        amigos[5] = "Karla";
        amigos[6] = "Ricardo";
        amigos[7] = "Mariela";
        amigos[8] = "Laura";
        amigos[9] = "Adriana";

        // Ubicar amigo
        int p = -1;
        for(int j=0; j<N; j++){
            if (amigos[j].equals(nombre)){
                p = j;
                break;
            }
        }

        // Imprimir lista de amigos
        System.out.println("Lista de Amigos\n");
        for(int j = 0; j < amigos.length; j++)
            System.out.println( j + "\t" + amigos[j] );

        // Imprimir resultado
        if (p == -1)
```

```
        System.out.println( "\n" + nombre + " no existe en la lista" );  
    else  
        System.out.println( "\n" + nombre + " esta en la posicion " + p );  
  
    System.out.println( "\n\t----- Fin -----" );  
  
    }  
}
```

El nombre del amigo a buscar lo debe ingresar tal como está en la lista, respetando mayúsculas y minúsculas.

Ejemplo 4

Este ejemplo muestra cómo hacer una búsqueda en un arreglo, se trata de una búsqueda secuencial.

Se tienen dos arreglos, el primero de ellos es **amigos** donde se tiene los nombres de amigos, y el segundo arreglo es **edades** donde se generan sus respectivas edades en forma aleatoria.

El programa se encarga de ubicar al amigo de mayor edad, para eso se asume que es el primero de la **lista**, con esta suposición se recorre el resto de la **lista** para comparar con cada uno de ellos y ubicar finalmente el amigo de mayor edad.

```
public class prog0704
{
    public static void main(String[] args)
    {
        int N = 10;
        String amigos[] = new String[N];
        int edades[] = new int[N];

        // Lista de amigos
        amigos[0] = "Sergio";
        amigos[1] = "Claudia";
        amigos[2] = "Hugo";
        amigos[3] = "Delia";
        amigos[4] = "Julio";
        amigos[5] = "Karla";
        amigos[6] = "Ricardo";
        amigos[7] = "Mariela";
        amigos[8] = "Laura";
        amigos[9] = "Adriana";

        // Generar edades
        for(int j = 0; j < edades.length; j++)
            edades[j] = (int)(Math.random()*100);

        // Ubicar al de mayor edad
        // Se asume que el de mayor edad es el primero de la lista
        int mayor = edades[0], p = 0;
        // Luego se hace un recorrido del arreglo
        for(int j =1; j < N; j++ ){
            if (mayor < edades[j]){
                mayor = edades[j];
            }
        }
    }
}
```

```
        p = j;
    }
}

// Imprimir lista de amigos
System.out.println("Lista de Amigos\n");
for(int j = 0; j < amigos.length; j++){
    System.out.println( j + "\t" + amigos[j] + "\t" + edades[j] );
}

// Imprimir el de mayor edad
System.out.println("\nResultado de la Búsqueda\n");
System.out.println("Posicion: " + p);
System.out.println("Amigo: " + amigos[p]);
System.out.println("Edad: " + edades[p]);

System.out.println( "\n\t----- Fin -----" );

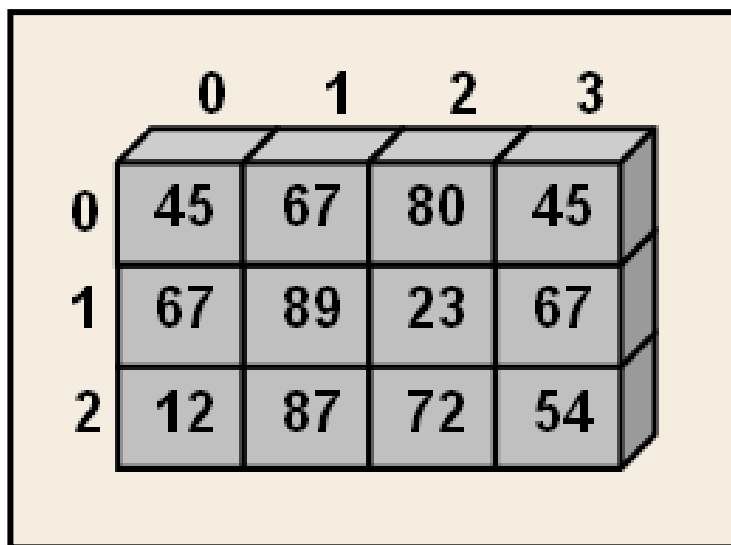
}
}
```

ARREGLOS BIDIMENSIONALES – MATRICES

Podemos considerar la siguiente situación:

“Si cada elemento de un arreglo unidimensional es otro arreglo, entonces tenemos un arreglo de dos dimensiones, si los elementos de este segundo arreglo son otros arreglos, entonces tenemos un arreglo de tres dimensiones, y así sucesivamente.”

Un arreglo bidimensional se representa por un conjunto de filas y columnas, tal como se muestra en la Figura 3, para acceder a un elemento en particular se necesita conocer la fila y columna donde se encuentra.



	0	1	2	3
0	45	67	80	45
1	67	89	23	67
2	12	87	72	54

Figura 3 Representación de un arreglo bidimensional.

Declaración de Arreglos Bidimensionales

Sintaxis

```
tipo variable_arreglo[][];
```

o

```
tipo[][] variable_arreglo;
```

A continuación, tienes algunos ejemplos:

```
int mat1[][];  
String mat2[][];
```

Debes tener presente que solo estás creando la variable que va a contener el arreglo, pero no el arreglo en sí.

Creación de Arreglos

Para la creación del arreglo debemos utilizar el operador new.

Sintaxis

```
variable_arreglo = new tipo[filas][columnas];
```

Es necesario haber declarado la variable **variable_arreglo** previamente.

Veamos algunos ejemplos:

```
mat1 = new int[10][5];  
mat2 = new String[5][10];  
  
int mat3[][] = new int[4][];  
mat3[0] = new int[5];  
mat3[1] = new int[5];  
mat3[2] = new int[5];  
mat3[3] = new int[5];
```

Acceso a los Elementos de una Matriz

Para acceder a los elementos de una matriz necesitamos conocer la fila y columna donde está ubicado el elemento.

Sintaxis

```
variable_arreglo[fila][columna]
```

Por ejemplo, para guardar en la fila 2, columna 3 de la matriz **mat1** el valor 20, la instrucción es:

```
mat1[2][3] = 20;
```

Para imprimir el contenido de **mat1**, las instrucciones son:

```
for( int i = 0; i < mat.length; i++ )
{
    for( int j = 0; j < mat[i].length; j++ )
    {
        System.out.print( mat[i][j] + "\t");
    }
    System.out.println();
}
```

Inicialización de una Matriz

Podemos crear e inicializar una matriz al mismo tiempo.

Sintaxis

```
tipo variable_arreglo[][] = {  
    {elemento,elemento,...},  
    {elemento,elemento,...}  
};
```

Veamos el siguiente ejemplo:

```
int[][] mat = { {13,15,18}, {67,23,56}, {15,45,23} };  
  
for( int i = 0; i < mat.length; i++ )  
{  
    for( int j = 0; j < mat[i].length; j++ )  
    {  
        System.out.print( mat[i][j] + "\t");  
    }  
    System.out.println();  
}
```

APLICACIÓN DE ARREGLOS BIDIMENSIONALES

Ejemplo 5

Este ejemplo trata de la suma de matrices, para lo cual es necesario que ambas matrices tengan las mismas dimensiones.

Las matrices se generan de manera aleatoria.

Para imprimir las matrices se ha creado el método **sayMatriz**, que se encarga de imprimir una matriz que se le pasa como parámetro.

```
public class prog0705
{
    public static void main(String[] args)
    {
        int mat1[][] = new int[4][5];
        int mat2[][] = new int[4][5];
        int suma[][] = new int[4][5];

        // Generar Matriz 1
        for( int i=0; i<mat1.length; i++ )
            for( int j=0; j<mat1[i].length; j++ )
                mat1[i][j] = (int)(Math.random()*20 + 10);

        // Generar Matriz 2
        for( int i=0; i<mat2.length; i++ )
            for( int j=0; j<mat2[i].length; j++ )
                mat2[i][j] = (int)(Math.random()*20 + 10);

        // Calcular Suma
        for( int i=0; i<mat2.length; i++ )
            for( int j=0; j<mat2[i].length; j++ )
                suma[i][j] = mat1[i][j] + mat2[i][j];

        // Imprimir Matrices
        sayMatriz( "\nMatriz 1", mat1 );
        sayMatriz( "\nMatriz 2", mat2 );
        sayMatriz( "\nMatriz Suma", suma );
    }
}
```

```
private static void sayMatriz( String titulo, int mat[][] ){
    System.out.println( titulo + "\n" );
    for( int i = 0; i < mat.length; i++ )
    {
        for( int j = 0; j < mat[i].length; j++ )
            System.out.print( mat[i][j] + "\t");
        System.out.println("");
    }
}
}
```

Ejemplo 6

Este ejemplo trata del producto de dos matrices, para poder multiplicar dos matrices se debe cumplir:

$$\text{prod}_{ij} = \text{mat1}_{ik} * \text{mat2}_{kj}$$

Por lo tanto, para encontrar **prod[3][4]**, se opera con la **fila 3** de la **mat1** y con la **columna 4** de **mat2**, suponiendo que son 2 columnas de mat1, por lo tanto deben ser 2 filas de mat2, la operación sería así:

```
prod[3][4] = mat1[3][1] * mat2[1][4] + mat1[3][2] * mat2[2][4];
```

Aplicando este mismo razonamiento se calcula todos los elementos de la matriz producto, lo más recomendable es hacerlo en un bucle, por ejemplo, un **for**, debes tener en cuenta que el índice inicia en cero (0), tal como se ilustra en el programa.

```
public class prog0706
{
    public static void main(String[] args)
    {
        int mat1[][] = new int[4][2];
        int mat2[][] = new int[2][3];
        int prod[][] = new int[4][3];

        // Generar Matriz 1
        for( int i=0; i<mat1.length; i++ )
            for( int j=0; j<mat1[i].length; j++ )
                mat1[i][j] = (int)(Math.random()*10);

        // Generar Matriz 2
        for( int i=0; i<mat2.length; i++ )
            for( int j=0; j<mat2[i].length; j++ )
                mat2[i][j] = (int)(Math.random()*10);

        // Calcular Producto
        for( int i=0; i<prod.length; i++ )
            for( int j=0; j<prod[i].length; j++ )
            {
                int suma = 0;
```

```
        for( int k = 0; k<2; k++ )
            suma += mat1[i][k] * mat2[k][j];
        prod[i][j] = suma;
    }

    // Imprimir Matrices
    sayMatriz( "\nMatriz 1", mat1 );
    sayMatriz( "\nMatriz 2", mat2 );
    sayMatriz( "\nMatriz Producto", prod );

    System.out.println( "\n\t----- Fin -----" );

}

private static void sayMatriz( String titulo, int mat[][] ){
    System.out.println( titulo + "\n" );
    for( int i = 0; i < mat.length; i++ )
    {
        for( int j = 0; j < mat[i].length; j++ )
            System.out.print( mat[i][j] + "\t");
        System.out.println("");
    }
}
}
```

CURSOS RELACIONADOS

<https://www.ceps.uni.edu.pe/>



