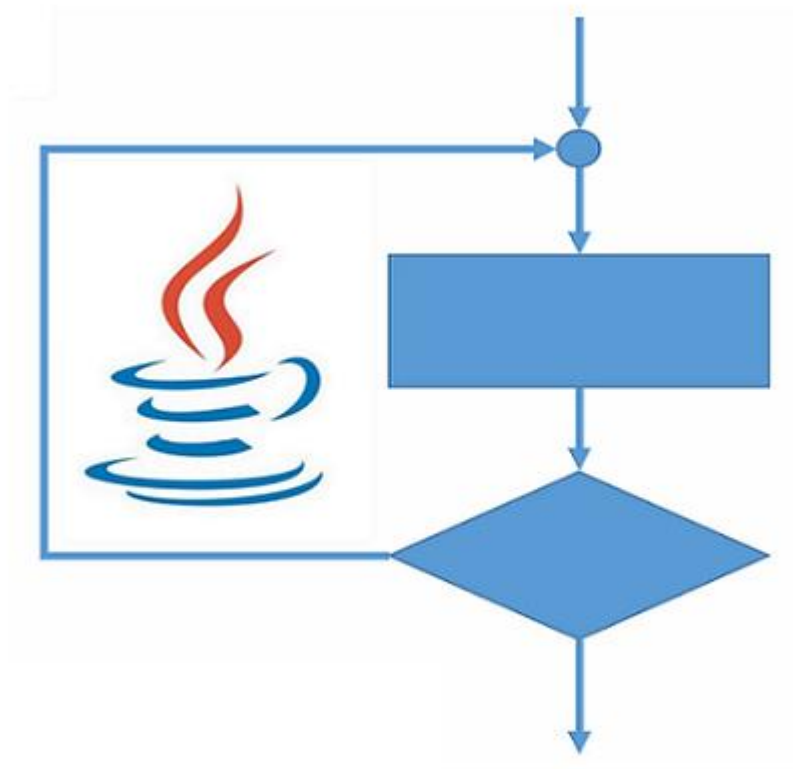




GUSTAVO CORONEL
DESARROLLA SOFTWARE

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.....	10
EJEMPLO 3.....	13
EJEMPLO 4.....	16
ARREGLOS BIDIMENSIONALES – MATRICES	19
DECLARACIÓN DE ARREGLOS BIDIMENSIONALES.....	20
CREACIÓN DE ARREGLOS	21
ACCESO A LOS ELEMENTOS DE UNA MATRIZ.....	22
INICIALIZACIÓN DE UNA MATRIZ	23
APLICACIÓN DE ARREGLOS BIDIMENSIONALES	24
EJEMPLO 5.....	24
EJEMPLO 6.....	27
CURSOS VIRTUALES	30
CUPONES.....	30
FUNDAMENTOS DE PROGRAMACIÓN CON JAVA	30
JAVA ORIENTADO A OBJETOS	31
PROGRAMACIÓN CON JAVA JDBC	32
PROGRAMACIÓN CON ORACLE PL/SQL.....	33

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.

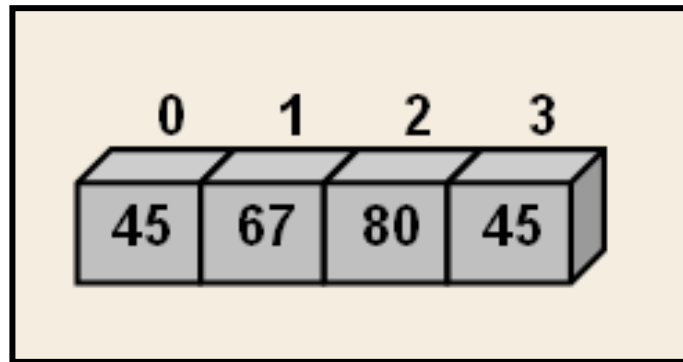


Figura 2 Arreglo Unidimensional

Declaración de Arreglos

Sintaxis

```
tipo variable_arreglo[];
```

o

```
tipo[] variable_arreglo;
```

En este caso solo estas creando la variable que apuntara al 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 lee con un objeto `Scanner`, 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.

```
import java.util.Scanner;

/**
 * @author Eric Gustavo Coronel Castillo
 * @blog www.desarrollasoftware.com
 * @email gcoronelc@gmail.com
 * @youtube www.youtube.com/DesarrollaSoftware
 * @facebook www.facebook.com/groups/desarrollasoftware
 * @cursos gcoronelc.github.io
 */
public class Ejemplo01 {

    public static void main(String[] args) {
        // Variables
        Scanner scanner = new Scanner(System.in);
        int n, lista[], lista2[], temp;
        // Leer tamaño de la lista
        do {
            System.out.print("Ingrese tamaño de la lista: ");
            n = scanner.nextInt();
            if (n < 5) {
                System.err.println("ERROR: El tamaño debe ser mínimo 5.");
            }
        }
```



```
} while (n < 5);
// Creación de arreglos
lista = new int[n]; // Lista original
// Generar Lista
for (int k = 0; k < n; k++) {
    lista[k] = (int) (Math.random() * 20 + 1);
}
// Hacer una copia de lista en lista2
lista2 = lista.clone();
// Ordenar Lista - Método Burbuja
for (int i = 0; i < (n - 1); i++) {
    for (int j = i + 1; j < n; j++) {
        if (lista2[i] > lista2[j]) {
            temp = lista2[i];
            lista2[i] = lista2[j];
            lista2[j] = temp;
        }
    }
}
// Imprimir lista ordenada
System.out.println("");
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 -----");
}
}
```




Ejecución:

```
run:
Ingrese tamaño de la lista: 3
ERROR: El tamaño debe ser mínimo 5.
Ingrese tamaño de la lista: 6

Lista Generada      Lista Ordenada
      19                2
      15                3
       3                9
       9               14
      14               15
       2               19

----- 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.

```
import java.util.Scanner;

/**
 * @author Eric Gustavo Coronel Castillo
 * @blog www.desarrollasoftware.com
 * @email gcoronelc@gmail.com
 * @youtube www.youtube.com/DesarrollaSoftware
 * @facebook www.facebook.com/groups/desarrollasoftware
 * @cursos gcoronelc.github.io
 */
public class Ejemplo02 {

    public static void main(String[] args) {
        // Variables
        Scanner scanner = new Scanner(System.in);
        int n, lista[], p;
        double mediana;
        // Leer el tamaño del arreglo
        do{
            System.out.print("Ingrese tamaño del arreglo: ");
            n = scanner.nextInt();
            if(n<5){
                System.err.println("ERROR: El tamaño debe ser mínimo 5.");
            }
        }while(n<5);
        // Generar Lista
        lista = new int[n];
        for (int k = 0; k < n; k++) {
            lista[k] = (int) (Math.random() * 100 + 1);
        }
    }
}
```



```
// Ordenar Lista - Método 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
p = n / 2; // Punto central
if (n%2 == 0) {
    mediana = (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 -----");
}

}
```



Ejecución:

```
run:
Ingrese tamaño del arreglo: 4
ERROR: El tamaño debe ser mínimo 5.
Ingrese tamaño del arreglo: 6
Lista Generada
    1
    3
    7
   23
   57
   61
Mediana: 15.0

----- 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 se está utilizando la clase [Scanner](#).

```
import java.util.Scanner;

/**
 * @author Eric Gustavo Coronel Castillo
 * @blog www.desarrollasoftware.com
 * @email gcoronelc@gmail.com
 * @youtube www.youtube.com/DesarrollaSoftware
 * @facebook www.facebook.com/groups/desarrollasoftware
 * @cursos gcoronelc.github.io
 */
public class Ejemplo03 {

    public static void main(String[] args) {
        // Variables
        final int N = 10;
        String amigos[] = new String[N];
        String nombre;
        Scanner scanner = new Scanner(System.in);
        char rpt;
        // Lista de amigos
        amigos[0] = "Sergio";
        amigos[1] = "Claudia";
        amigos[2] = "Guino";
        amigos[3] = "Delia";
        amigos[4] = "Gustavo";
        amigos[5] = "Karla";
        amigos[6] = "Ricardo";
        amigos[7] = "Mariela";
        amigos[8] = "Ernesto";
        amigos[9] = "Adriana";
        // Bucle de control
        do {
            // Leer nombre del amigo
```



```
System.out.println("BUSCAR TU AMIGO");
System.out.print("Nombre de tu amigo: ");
nombre = scanner.nextLine().toUpperCase();
// Ubicar amigo
int p = -1;
for (int j = 0; j < N; j++) {
    if (amigos[j].toUpperCase().equals(nombre)) {
        p = j;
        break;
    }
}
// Reporte
System.out.println("");
System.out.println("REPORTE");
if (p == -1) {
    System.out.println(nombre + " no existe en la lista.");
} else {
    System.out.println(nombre + " esta en la posicion " + p + ".");
}
// Otra busqueda
System.out.println("");
System.out.print("Realiza otra busqueda (S/N): ");
rpta = scanner.next().toUpperCase().charAt(0);
scanner.nextLine();
System.out.println("");
} while (rpta == 'S');
System.out.println("----- Fin -----");
}
}
```

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



Ejecución:

```
run:
BUSCAR TU AMIGO
Nombre de tu amigo: Ernesto

REPORTE
ERNESTO esta en la posicion 2.

Realiza otra busqueda (S/N): S

BUSCAR TU AMIGO
Nombre de tu amigo: Claudia

REPORTE
CLAUDIA esta en la posicion 1.

Realiza otra busqueda (S/N): N

----- Fin -----
```



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.

```
/**
 * @author Eric Gustavo Coronel Castillo
 * @blog www.desarrollasoftware.com
 * @email gcoronelc@gmail.com
 * @youtube www.youtube.com/DesarrollaSoftware
 * @facebook www.facebook.com/groups/desarrollasoftware
 * @cursos gcoronelc.github.io
 */
public class Ejemplo04 {

    public static void main(String[] args) {
        // Variables
        final int N = 10; // Tamaño del arreglo
        String amigos[] = new String[N];
        int edades[] = new int[N];
        int mayorEdad, indiceMayorEdad;
        // Lista de amigos
        amigos[0] = "Sergio";
        amigos[1] = "Claudia";
        amigos[2] = "Ernesto";
        amigos[3] = "Delia";
        amigos[4] = "Gustavo";
        amigos[5] = "Karla";
        amigos[6] = "Ricardo";
        amigos[7] = "Mariela";
        amigos[8] = "Laura";
        amigos[9] = "Adriana";
    }
}
```




```
// Generar edades
for (int i = 0; i < edades.length; i++) {
    edades[i] = (int) (Math.random() * 50);
}

// Ubicar al de mayor edad
// Se asume que el de mayor edad es el primero de la lista
mayorEdad = edades[0];
indiceMayorEdad = 0;

// Luego se hace un recorrido del arreglo
for (int i = 1; i < N; i++) {
    if (mayorEdad < edades[i]) {
        mayorEdad = edades[i];
        indiceMayorEdad = i;
    }
}

// Imprimir lista de amigos
System.out.println("LISTA DE AMIGOS");
System.out.println("=====");
for (int i = 0; i < N; i++) {
    System.out.println(i + "\t" + amigos[i] + "\t" + edades[i]);
}

// Imprimir el de mayor edad
System.out.println("");
System.out.println("RESULTADO DE LA BUSQUEDA");
System.out.println("=====");
System.out.println("Indice: " + indiceMayorEdad);
System.out.println("Amigo: " + amigos[indiceMayorEdad]);
System.out.println("Edad: " + edades[indiceMayorEdad]);
System.out.println("\n----- Fin -----");
}
}
```



Ejecución:

```
run:
LISTA DE AMIGOS
=====
0      Sergio 2
1      Claudia      43
2      Ernesto      12
3      Delia 29
4      Gustavo      42
5      Karla 36
6      Ricardo      3
7      Mariela      27
8      Laura 37
9      Adriana      40

RESULTADO DE LA BUSQUEDA
=====
Indice: 1
Amigo: Claudia
Edad: 43

----- Fin -----
```

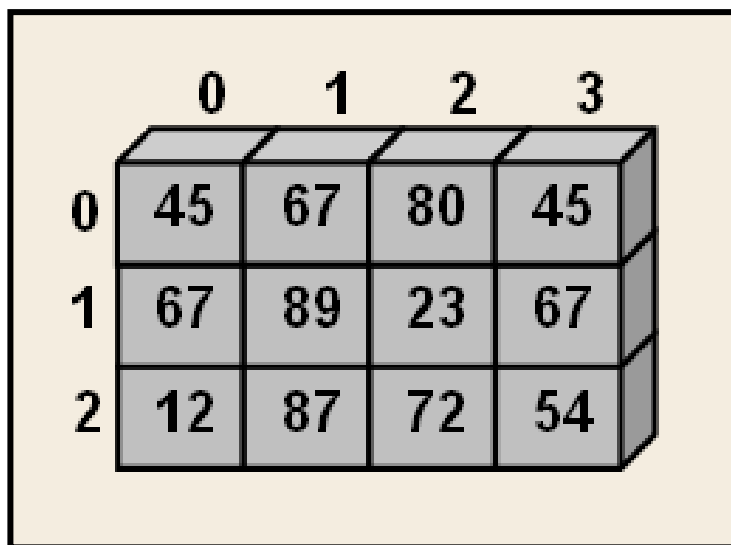


ARREGLOS BIDIMENSIONALES – MATRICES

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 apuntara al 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 es necesario 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

Puedes 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. En este ejemplo se están utilizando 2 métodos:

- generarMatriz** Este método se encarga de generar una matriz con datos aleatorios. Se le pasa como parámetros el tamaño de la matriz.
- mostrarMatriz** Este método se encarga de mostrar una matriz. Recibe como parámetro la referencia a la matriz.

A continuación, tienes el programa completo.

```
/**
 * @author Eric Gustavo Coronel Castillo
 * @blog www.desarrollasoftware.com
 * @email gcoronelc@gmail.com
 * @youtube www.youtube.com/DesarrollaSoftware
 * @facebook www.facebook.com/grupos/desarrollasoftware
 * @cursos gcoronelc.github.io
 */
public class Ejemplo05 {

    public static void main(String[] args) {
        // Constantes
        final int FILAS = 3;
        final int COLUMNAS = 4;
        // Variables
        int mat1[][], mat2[][], suma[][];
        // Generar Matrices
        mat1 = generarMatriz(FILAS, COLUMNAS);
        mat2 = generarMatriz(FILAS, COLUMNAS);
        // Calcular Suma
        suma = new int[FILAS][COLUMNAS];
        for (int i = 0; i < FILAS; i++) {
            for (int j = 0; j < COLUMNAS; j++) {
```




```
        suma[i][j] = mat1[i][j] + mat2[i][j];
    }
}

// Imprimir Matrices
mostrarMatriz("\nMatriz 1", mat1);
mostrarMatriz("\nMatriz 2", mat2);
mostrarMatriz("\nMatriz Suma", suma);
System.out.println("\n----- Fin -----");
}

/**
 * Muestra una matriz en la consola.
 * @param titulo Titulo de la matriz.
 * @param mat Matriz con datos a mostrar.
 */
private static void mostrarMatriz(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("");
    }
}

/**
 * Genera una matriz con datos enteros.
 * @param filas Cantidad de filas de la matriz.
 * @param columnas Cantidad de columnas de la matriz.
 * @return Retorna la referencia a la matriz generada.
 */
private static int[][] generarMatriz(int filas, int columnas) {
    int mat[][] = new int[filas][columnas];
    for (int i = 0; i < filas; i++) {
        for (int j = 0; j < columnas; j++) {
            mat[i][j] = (int) (Math.random() * 20 + 10);
        }
    }
}
```



```
    return mat;  
}  
  
}
```

Ejecución:

run:

Matriz 1

14	28	20	28
21	10	13	15
20	17	24	29

Matriz 2

24	28	22	15
16	21	13	14
25	19	28	17

Matriz Suma

38	56	42	43
37	31	26	29
45	36	52	46

----- Fin -----



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.

```
/**
 * @author Eric Gustavo Coronel Castillo
 * @blog www.desarrollasoftware.com
 * @email gcoronelc@gmail.com
 * @youtube www.youtube.com/DesarrollaSoftware
 * @facebook www.facebook.com/groups/desarrollasoftware
 * @cursos gcoronelc.github.io
 */
public class Ejemplo06 {

    public static void main(String[] args) {
        // Variables
        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 Matriz 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;
    }
}
// Mostrar Matrices
mostrarMatriz("\nMatriz 1", mat1);
mostrarMatriz("\nMatriz 2", mat2);
mostrarMatriz("\nMatriz Producto", prod);
System.out.println("\n----- Fin -----");
}

/**
 * Muestra una matriz en la consola.
 *
 * @param titulo Titulo de la matriz.
 * @param mat Matriz con datos a mostrar.
 */
private static void mostrarMatriz(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("\t" + mat[i][j]);
        }
        System.out.println("");
    }
}
```



```
}  
  
}
```

Ejecución:

```
run:
```

```
Matriz 1
```

```
    9    5  
    7    1  
    1    8  
    4    6
```

```
Matriz 2
```

```
    3    6    5  
    9    3    7
```

```
Matriz Producto
```

```
    72    69    80  
    30    45    42  
    75    30    61  
    66    42    62
```

```
----- Fin -----
```

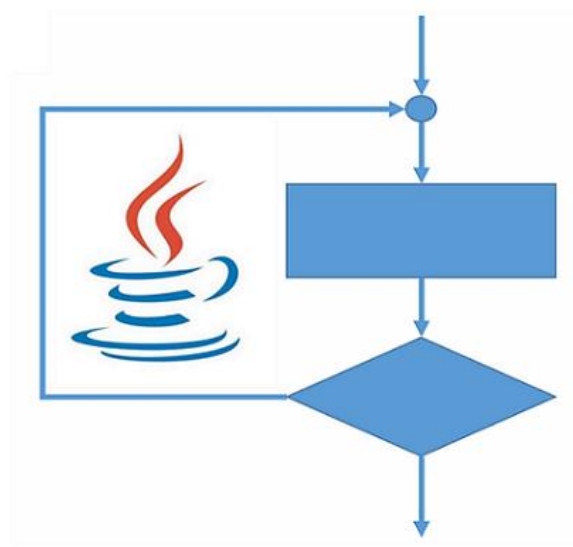
CURSOS VIRTUALES

CUPONES

En esta URL se publican cupones de descuento:

<http://gcoronelc.github.io>

FUNDAMENTOS DE PROGRAMACIÓN CON JAVA



Tener bases sólidas de programación muchas veces no es fácil, creo que es principalmente por que en algún momento de tu aprendizaje mezclas la entrada de datos con el proceso de los mismos, o mezclas el proceso con la salida o reporte, esto te lleva a utilizar malas prácticas de programación que luego te serán muy difíciles de superar.

En este curso aprenderás las mejores prácticas de programación para que te inicies con éxito en este competitivo mundo del desarrollo de software.

URL del Curso: **<https://www.udemy.com/course/fund-java>**

Avance del curso: **<https://n9.cl/gcoronelc-fp-avance>**

Cupones de descuento: **<http://gcoronelc.github.io>**



JAVA ORIENTADO A OBJETOS



CURSO PROFESIONAL DE JAVA ORIENTADO A OBJETOS

Eric Gustavo Coronel Castillo

www.desarrollasoftware.com

I N S T R U C T O R

En este curso aprenderás a crear software aplicando la Orientación a Objetos, la programación en capas, el uso de patrones de software y Swing.

Cada tema está desarrollado con ejemplos que demuestran los conceptos teóricos y finalizan con un proyecto aplicativo.

URL del Curso: **<https://bit.ly/2B3ixUW>**

Avance del curso: **<https://bit.ly/2RYGXIt>**

Cupones de descuento: **<http://gcoronelc.github.io>**

PROGRAMACIÓN CON JAVA JDBC



PROGRAMACIÓN DE BASE DE DATOS ORACLE CON JAVA JDBC

Eric Gustavo Coronel Castillo

www.desarrollasoftware.com

I N S T R U C T O R

En este curso aprenderás a programar bases de datos Oracle con JDBC utilizando los objetos Statement, PreparedStatement, CallableStatement y a programar transacciones correctamente teniendo en cuenta su rendimiento y concurrencia.

Al final del curso se integra todo lo desarrollado en una aplicación de escritorio.

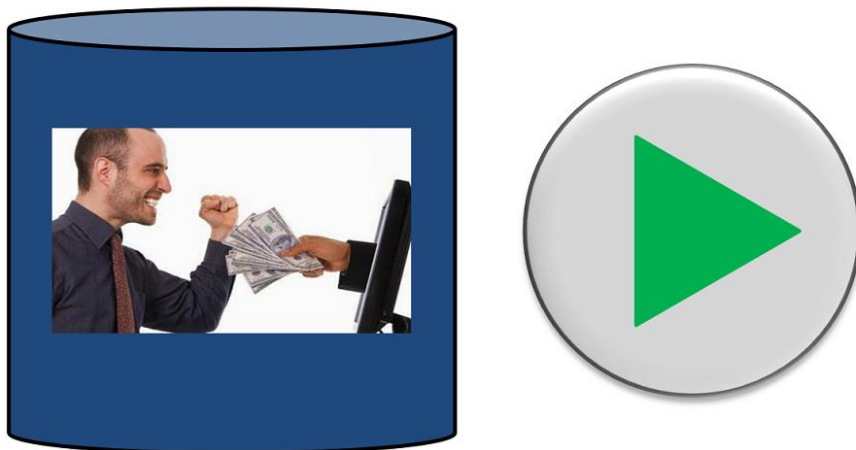
URL del Curso: <https://bit.ly/31apy00>

Avance del curso: <https://bit.ly/2vatZOT>

Cupones de descuento: <http://gcoronelc.github.io>

PROGRAMACIÓN CON ORACLE PL/SQL

ORACLE PL/SQL



En este curso aprenderás a programar las bases de datos ORACLE con PL/SQL, de esta manera estarás aprovechando las ventajas que brinda este motor de base de datos y mejorarás el rendimiento de tus consultas, transacciones y la concurrencia.

Los procedimientos almacenados que desarrolles con PL/SQL se pueden ejecutarlos de Java, C#, PHP y otros lenguajes de programación.

URL del Curso: <https://bit.ly/2YZjfxT>

Avance del curso: <https://bit.ly/3bcqYb>

Cupones de descuento: <http://gcoronelc.github.io>