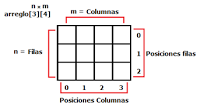
**Matrices**

Este tipo de arreglos son conocidos como matrices y corresponden a una estructura de datos que puede almacenar muchos más datos que los arreglos unidimensionales, pues estos últimos como ya se mencionó se componen de una fila por n columnas, mientras que los bidimensionales se componen de n filas por m columnas.



int matriz[][] = **new** int[3][3];

for (int x=0; x < matriz.length; x++) {

for (int y=0; y < matriz[x].length; y++) {

[**System**](http://www.google.com/search?hl=en&q=allinurl%3ASystem+java.sun.com&btnI=I%27m%20Feeling%20Lucky).out.println (matriz[x][y]);

}

}

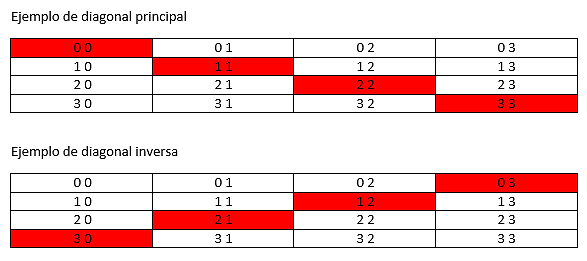
1. Crear una matriz de 3×3 con los números del 1 al 9. Mostrar por pantalla, tal como aparece en la matriz.
2. Crear una matriz de 5 filas y n columnas (se pide al usuario). Rellenarlo con números aleatorios entre 0 y 10.
3. Crear dos matrices de nxn y sumar sus valores, los resultados se deben almacenar en otra matriz. Los valores y la longitud, seran insertados por el usuario. Mostrar las matrices originales y el resultado.
4. Programa que genera una matriz representando un triángulo invertido alineado a la derecha realizado con asteriscos, cuyo tamaño será especificado por el usuario. La matriz será mostrada en pantalla finalmente. Por ejemplo, si se indica el tamaño 5 deberá aparecer:

\*\*\*\*\*  
\*\*\*\*  
\*\*\*  
\*\*  
\*

1. Nos piden crear una matriz de 4×4 de números enteros que inicialmente esta vacía, nos piden hacer un menú con estas opciones:

* Rellenar TODA la matriz de números, debes pedírselo al usuario.
* Suma de una fila que se pedirá al usuario (controlar que elija una correcta)
* Suma de una columna que se pedirá al usuario (controlar que elija una correcta)
* Sumar la diagonal principal (ver ejemplo)
* Sumar la diagonal inversa (ver ejemplo)
* La media de todos los valores de la matriz

IMPORTANTE: hasta que no se haga la primera opción, el resto de opciones no se deberán de ejecutar, simplemente mostrar un mensaje donde diga que debes rellenar la matriz.



1. Nos piden que implementemos el software para una máquina expendedora de golosinas.

Cada golosina tiene un nombre y un precio para ahorrar tiempo, se pasa los datos que tendrá a continuación (copiar y pegar):

String[][] nombresGolosinas = {

{"KitKat", "Chicles de fresa", "Lacasitos", "Palotes"},

{"Kinder Bueno", "Bolsa variada Haribo", "Chetoos", "Twix"},

{"Kinder Bueno", "M&M'S", "Papa Delta", "Chicles de menta"},

{"Lacasitos", "Crunch", "Milkybar", "KitKat"}

};

double[][] precio = {

{1.1, 0.8, 1.5, 0.9},

{1.8, 1, 1.2, 1},

{1.8, 1.3, 1.2, 0.8},

{1.5, 1.1, 1.1, 1.1}

};

También tendrán una cantidad inicial, que en principio será de 5.

Tendremos un pequeño menú con las siguientes opciones:

* Pedir golosina: pedirá la posición de la golosina que quiera. Esta máquina tiene golosinas en cada posición, identificados por su fila y columna, que será lo que introduzca el usuario al pedir una golosina, por ejemplo si el usuario teclea 20 significa que está pidiendo la golosina que está en la fila 2 columna 0. Cuando no haya más golosinas se le indicará al usuario. Solo puede pedir una golosina y supondremos que el usuario siempre tiene dinero al elegir. Recuerda de disminuir la cantidad la pedir.
* Mostrar golosinas: mostrara todas las golosinas disponibles. Mostrará el código que debe introducir el usuario, el nombre y el precio. La cantidad no se mostrará.
* Rellenar golosinas: esta es una función exclusiva de un técnico por lo que nos pedirá una contraseña, si el usuario escribe “MaquinaExpendedora2017” le pedirá la posición de la golosina y la cantidad.
* Apagar maquina: sale del programa, antes de salir mostrara las ventas totales durante la ejecución del programa.

1. Realizar un juego de batalla naval. Tener dos matrices de 10 x 10 y distribuir los barcos de manera vertical y horizontal.

Porta aviones: 6 posiciones. (1 unidad)

Submarino: 4 posiciones ( 2 unidades )

Destroyer: 3 posiciones( 2 unidades )