CARRERA DE INGENIERÍA DE SISTEMAS

Arrays Bidimensionales

ESTRUCTURA DE DATOS Y ALGORITMOS
ÁREA DE INGENIERÍA DE SOFTWARE

UNIVERSIDAD DE LIMA

TEMARIO

- Conceptos básicos
- Operaciones y análisis de complejidad.
- Matrices poco densas y sus representaciones.

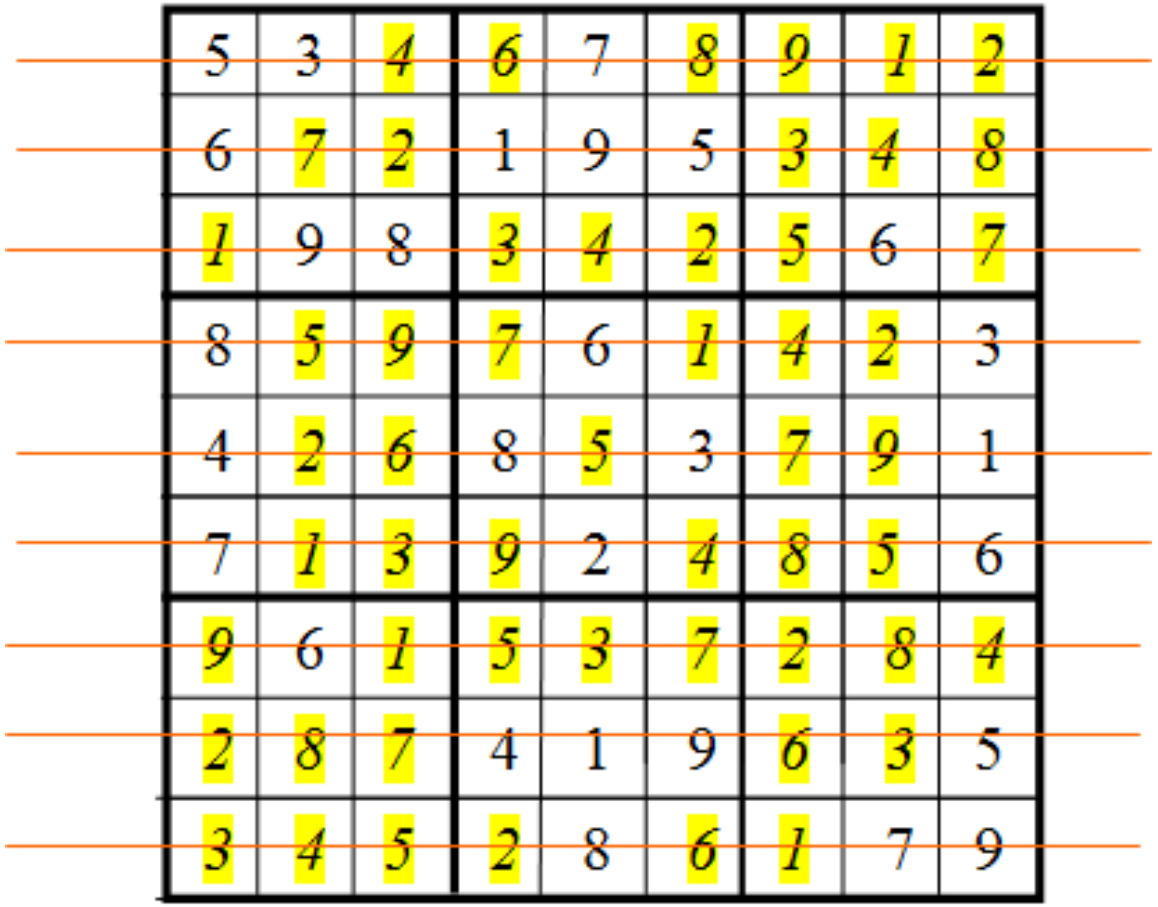
UNIVERSIDAD DE LIMA

¿Juegas sudoku?

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6							
			4	1	9			5
				8			7	9

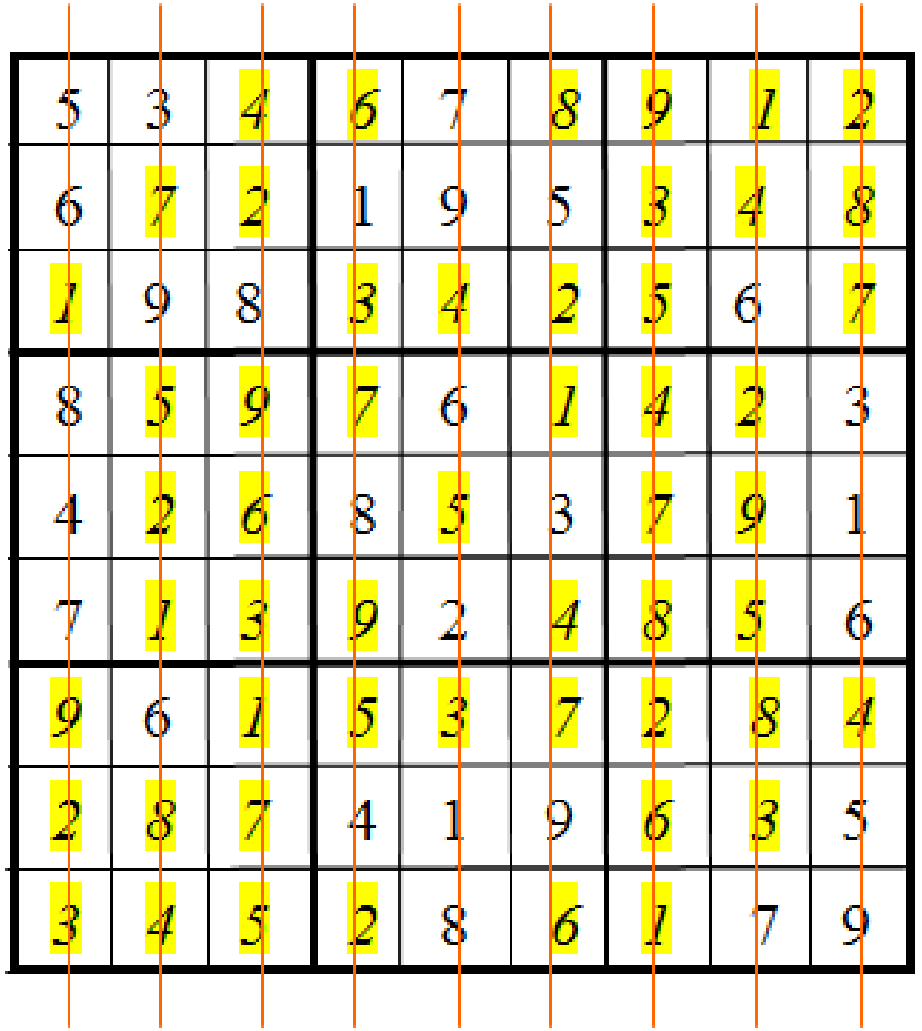
UNIVERSIDAD DE LIMA

En Sudoku cada fila debe contener los números del 1 al 9



UNIVERSIDAD DE LIMA

En Sudoku cada columna debe contener los números del 1 al 9



UNIVERSIDAD DE LIMA

En Sudoku cada recuadro de 3X3 debe contener los números del 1 al 9

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

UNIVERSIDAD DE LIMA

Verifique si la solución es correcta

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6							
			4	1	9			5
				8			7	9

5	3	*4*	*6*	7	*8*	*9*	*1*	*2*
6	*7*	*2*	1	9	5	*3*	*4*	*8*
1	9	8	*3*	*4*	*2*	*5*	6	*7*
8	*5*	*9*	*7*	6	*1*	*4*	*2*	3
4	*2*	*6*	8	*5*	3	*7*	*9*	1
7	*1*	*3*	*9*	2	*4*	*8*	*5*	6
9	6	*1*	*5*	*3*	*7*	*2*	*8*	*4*
2	*8*	*7*	4	1	9	*6*	*3*	5
3	*4*	*5*	*2*	8	*6*	*1*	7	9

UNIVERSIDAD DE LIMA

Array Bidimensional

- Un array bidimensional es un conjunto de datos homogéneo, finito y ordenado, donde se hace referencia a cada elemento por medio de dos índices.

- El primero se utiliza para indicar el renglón o fila y el segundo para indicar la columna.

- También puede definirse como un vector de vectores.

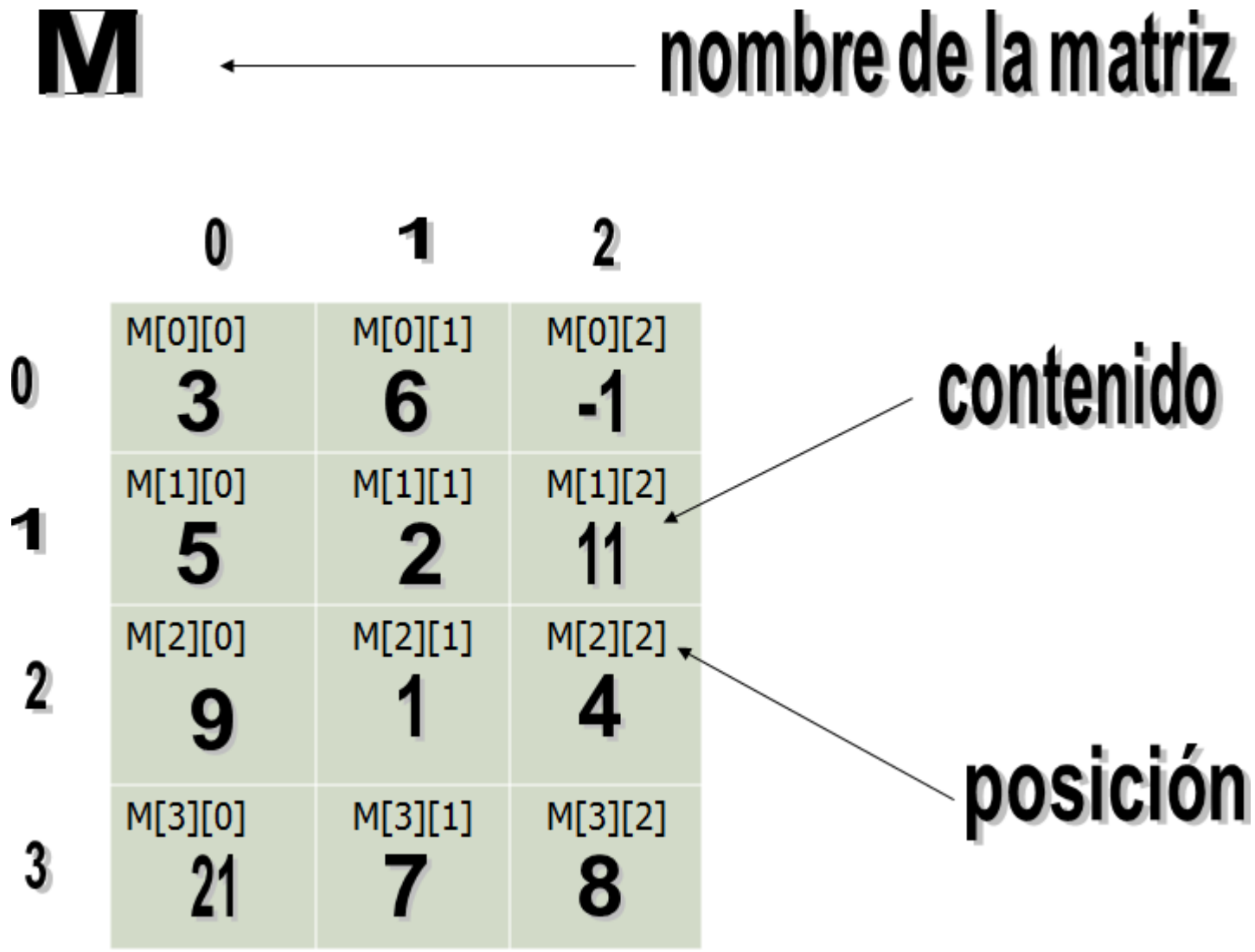
Matriz M(n x m)

M ← nombre de la Matriz

	0	1	2	.....	m
0	M[0][0]	M[0][1]	M[0][2]	...	M[0][m]
1	M[1][0]	M[1][1]	M[1][2]	...	M[1][m]
2	M[2][0]	M[2][1]	M[2][2]	...	M[2][m]
:					
n	M[n][0]	M[n][1]	M[n][2]	...	M[n][m]

UNIVERSIDAD DE LIMA

Matriz M(n x m)



UNIVERSIDAD DE LIMA

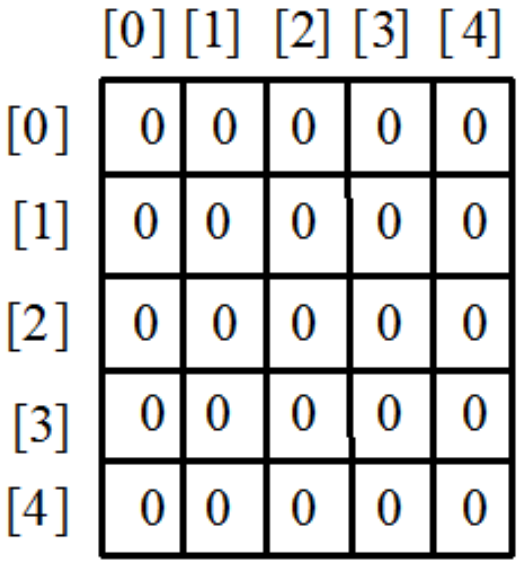
Declaración/Creación array bidimensional

```
// Declarar una matriz
dataType[][] matrix;

// Crear una matriz
matrix = new dataType[10][10];

// Combinar la declaración y creación de una matriz
dataType[][] matrix1 = new dataType[10][10];

// Otra alternativa
dataType matrix2[][] = new dataType[10][10];
```

Ejemplo de declaración y creación de matriz

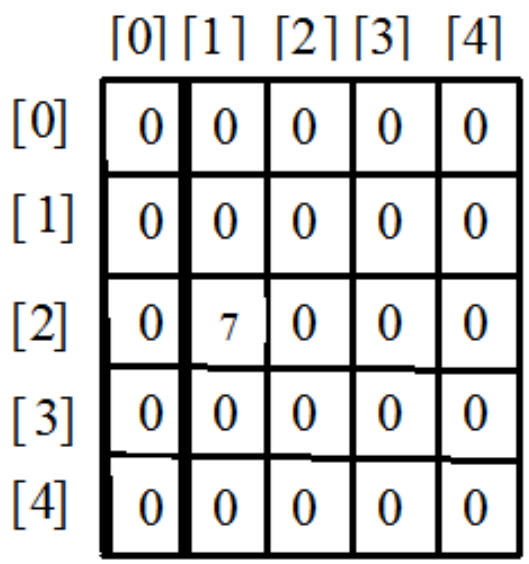
```
int[][] matrix = new int[10][10];
// Asignar un valor a una celda de la matriz
matrix[0][0] = 3;
// Crear una matriz con valores aleatorios
for (int i = 0; i < matrix.length; i++)
  for (int j = 0; j < matrix[i].length; j++)
    matrix[i][j] = (int)(Math.random() * 1000);
// Declarar una matriz de tipo double
double[][] x;
```

Ejemplo de declaración y creación de matriz



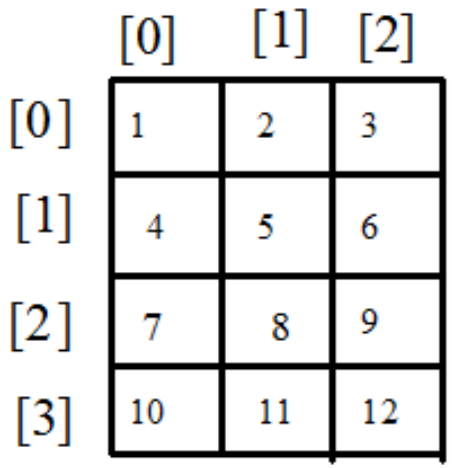
	[0]	[1]	[2]	[3]	[4]
[0]	0	0	0	0	0
[1]	0	0	0	0	0
[2]	0	0	0	0	0
[3]	0	0	0	0	0
[4]	0	0	0	0	0

```
matrix = new int[5][5];
```



	[0]	[1]	[2]	[3]	[4]
[0]	0	0	0	0	0
[1]	0	0	0	0	0
[2]	0	7	0	0	0
[3]	0	0	0	0	0
[4]	0	0	0	0	0

```
matrix[2][1] = 7;
```



	[0]	[1]	[2]
[0]	1	2	3
[1]	4	5	6
[2]	7	8	9
[3]	10	11	12

```
int[][] array = {
  {1, 2, 3},
  {4, 5, 6},
  {7, 8, 9},
  {10, 11, 12}
};
```

matrix.length?
5
matrix[0].length?
5

array.length?
4
array[0].length?
3

UNIVERSIDAD DE LIMA

Declarar, crear e inicializar usando notaciones abreviadas

También puede usar un inicializador de matriz para declarar, crear e inicializar una matriz bidimensional. Por ejemplo,

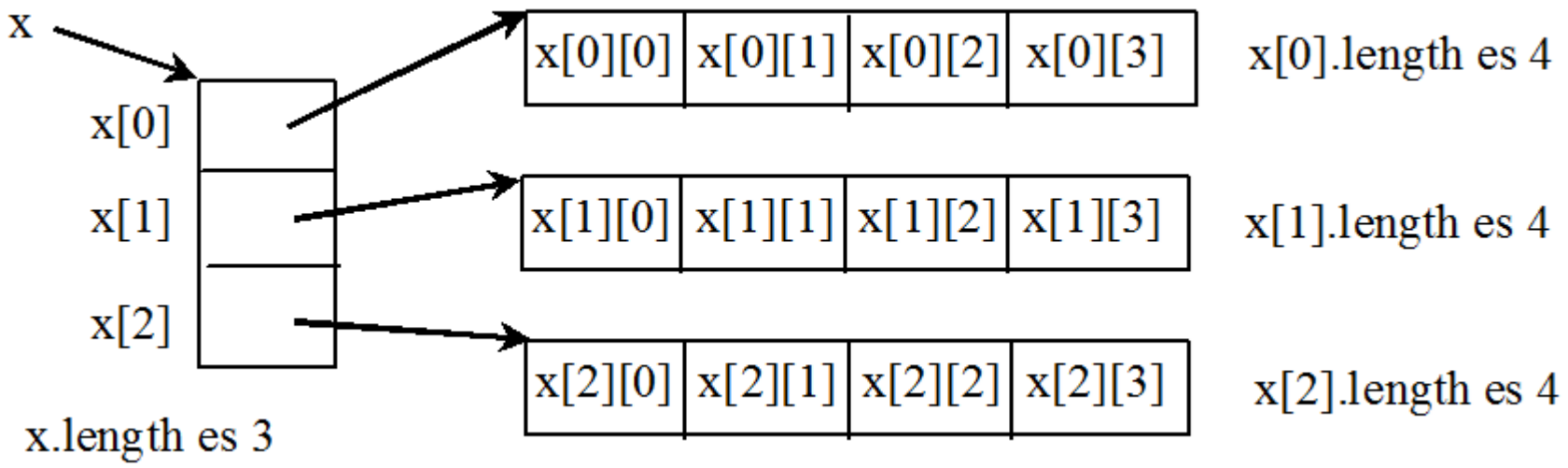
```
int[][] array = {
  {1, 2, 3},
  {4, 5, 6},
  {7, 8, 9},
  {10, 11, 12}
};
```

Similares

```
int[][] array = new int[4][3];
array[0][0] = 1; array[0][1] = 2; array[0][2] = 3;
array[1][0] = 4; array[1][1] = 5; array[1][2] = 6;
array[2][0] = 7; array[2][1] = 8; array[2][2] = 9;
array[3][0] = 10; array[3][1] = 11; array[3][2] = 12;
```

Longitud de los arrays bidimensionales

```
int[][] x = new int[3][4];
```



UNIVERSIDAD DE LIMA

Longitud de los arrays bidimensionales - Continua..

```
int[][] array = {
  {1, 2, 3},
  {4, 5, 6},
  {7, 8, 9},
  {10, 11, 12}
};
```

array.length	
array[0].length	
array[1].length	
array[2].length	
array[3].length	

array[4].length ArrayIndexOutOfBoundsException

UNIVERSIDAD DE LIMA

Operaciones con matrices

- ❑ Imprimir una Matriz
- ❑ Sumar todos los elementos de la matriz
- ❑ Sumando todos los elementos por columna

Operaciones propuestas:

- ❑ Qué fila tiene la suma más grande
- ❑ Encontrar el índice más pequeño del elemento más grande

UNIVERSIDAD DE LIMA

Imprimir una Matriz

```
for (int row = 0; fila < matrix.length; row++) {

  for (int column = 0; column < matrix[row].length; column++){

    System.out.print(matrix[row][column] + " ");

  }

  System.out.println();
}
```

UNIVERSIDAD DE LIMA

Sumar todos los elementos del array

```
int total = 0;

for (int row = 0; row < matrix.length; row++) {

  for (int column = 0; column < matrix[row].length; column++) {

    total += matrix[row][column];

  }

}
```

Sumar de los elementos por columna

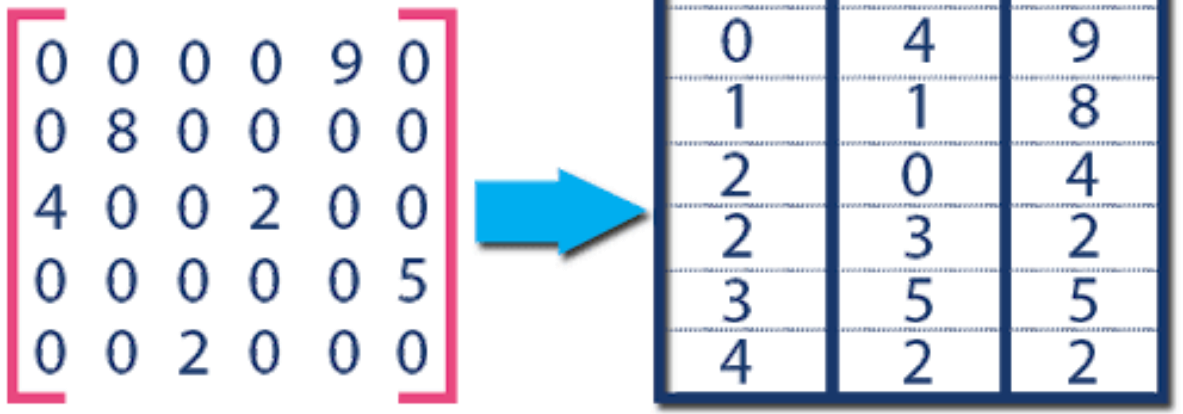
```
for (int column = 0; column < matrix[0].length; column++) {
  int total = 0;

  for (int row = 0; row < matrix.length; row++){
      total += matrix[row][column];
      System.out.println("Suma col. " + columna + " es " + total);
  }
}
```

UNIVERSIDAD DE LIMA

Matriz poco densa (sparse matrix)

Definición

Es aquella matriz con muy pocos elementos diferentes de cero. Sus características hacen que podamos desarrollar algoritmos para operar con ellas de forma eficiente.



0	0	0	0	9	0
0	8	0	0	0	0
4	0	0	2	0	0
0	0	0	0	0	5
0	0	2	0	0	0

Rows	Columns	Values
5	6	6
0	4	9
1	1	8
2	0	4
2	3	2
3	5	5
4	2	2

UNIVERSIDAD DE LIMA

Matriz poco densa (sparse matrix)

Especificaciones

Objetivo:	Almacenar los elementos de una matriz poco densa en un array lineal.
Entrada:	matriz A, filas, columnas
Precondición:	matriz A no vacía
Salida:	ans (array), idx
Postcondición:	idx representa la cantidad de elementos del array respuesta

Matriz poco densa (sparse matrix)

```
int idx = 0;
int[] ans = new int[1000];
for(int i = 0; i < rows; i++){
    for(int j = 0; j < columns; j++){
        if(A[i][j] != 0){
            ans[idx] = i;
            ans[idx + 1] = j;
            ans[idx + 2] = A[i][j];
            idx += 3;
        }
    }
}
```

UNIVERSIDAD DE LIMA

Referencias

1. Paul S. Wang, Java con programación orientada a objetos y aplicaciones en la WWW, México, 2000.
2. Sun microsystem, Fundamentals of the Java™ Programming Language SL-110-SE6
3. A.M. Vozmediano, Java para novatos, 2017.
4. BEGOÑA MOROS VALLE, http://dis.um.es/~bmoros/
5. http://lml.ls.fi.upm.es/ijava/ Ángel Lucas González Martínez

UNIVERSIDAD DE LIMA