Poo

https://github.com/NahuelArn

Lo q se ve

Tipos de clases q vemos

-Abstractas: son clases molde de las q nos extendemos con sus derivados ejemplo
Clase abstracta libro:
me extiendo con 2 clases concretas
-Libro Electronico
- Libro Fisico

- Concretas: se pueden instanciar(ejemplo el surtidor)

Polimorfismo
cuando los metodos los sobre escribimos, para que cambie su comportamiento

Herencia
capacidad de heredar de una clase los metodos y atributos del padre

Binding Dinamico
Se suele notar mas en clases abstractas, donde forzas el uso del this, para referirte a la clase actual (mirar video, de clase pre graba q lo explica joya)

Poo

https://github.com/NahuelArn

A mano

Randomize en Java

```java
import PaqueteLectura.GeneradorAleatorio; //o import scanner

public class Demo05Generador {

    public static void main(String[] args) {
        GeneradorAleatorio.iniciar();                           //inicia el generador aleatorio
        System.out.println(GeneradorAleatorio.generarInt(10));    //genera un int entre 0 y 9
        System.out.println(GeneradorAleatorio.generarDouble(10)); //genera un double entre 0 y 9
        System.out.println(GeneradorAleatorio.generarBoolean());  //genera un boolean
        System.out.println(GeneradorAleatorio.generarString(4));  //genera un string de 4 letras
        //Entre rangos
        System.out.println(GeneradorAleatorio.generarInt(1000)+7000);    //genera entre rangos 7000-8000
        System.out.println(GeneradorAleatorio.generarInt(loQueFaltaParaLlegarAlLimSuperior)+limiteInferior)
    }

}
```

lectura

```java
import PaqueteLectura.Lector; //scanner

public class Demo05Generador {

    public static void main(String[] args) {
        System.out.println("Ingrese un numero: ");
        int num1 = Lector.leerInt();
        System.out.println("Ingrese un nombre : ");
        String nombre = Lector.leerString();

         Lector.leerBoolean();
         Lector.leerDouble();

    }

}
```

Poo

Sobre carga

Polimorfismo
cuando los metodos los sobre escribimos, para que cambie su comportamiento

Herencia
capacidad de heredar de una clase los metodos y atributos del padre

Binding Dinamico
Se suele notar mas en clases abstractas, donde forzas el uso del this, para referirte a una c

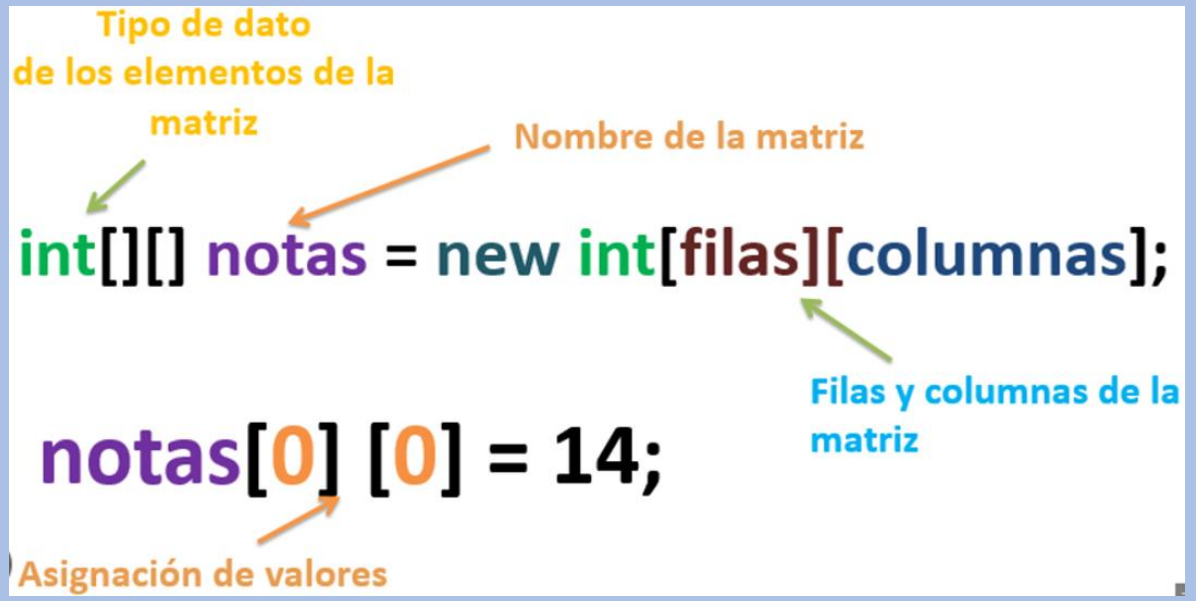
Poo

Si tengo only clases concretas(como la del surtidor)
-Escalera descendente, del surtidor hacia las clases contendidas

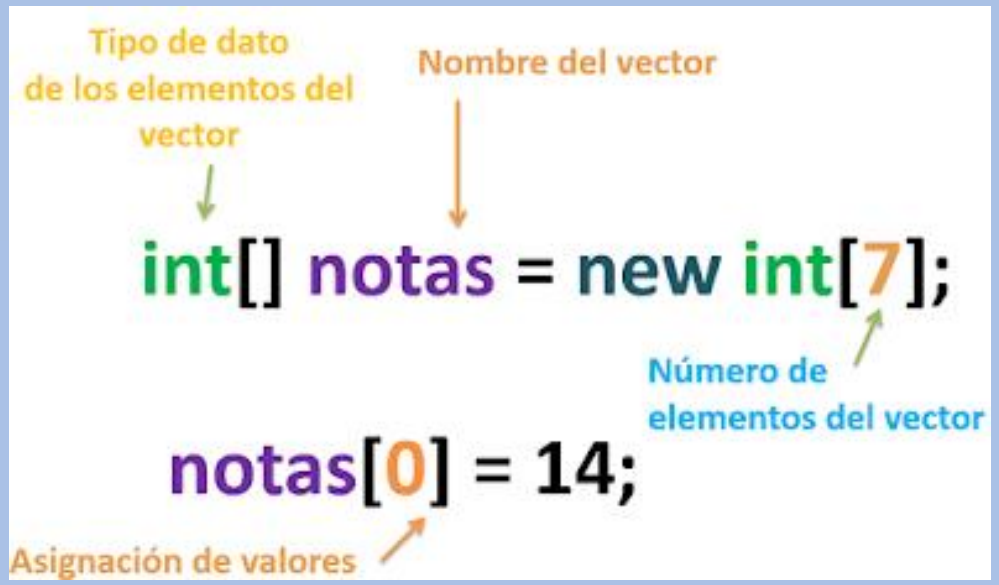
Si tengo clases abstractas es mas comun tener que extenderme con clases concretas y instanciar desde ellas,
-Escalera ascent(super, accedo a algunas cosas miran arriba)

Matrices

https://github.com/NahuelArn



Vector



Tipo de cargas en estructuras

-Si es de manera secuencial no me importa inicialiar la estructura cada campo

-Si se carga por posiciones x, es necesario inicializar la estructura con su valor por defecto, si es un objeto lo q contiene en null o si es entero en 0, etc etc…

Partidardo, vector N posicion

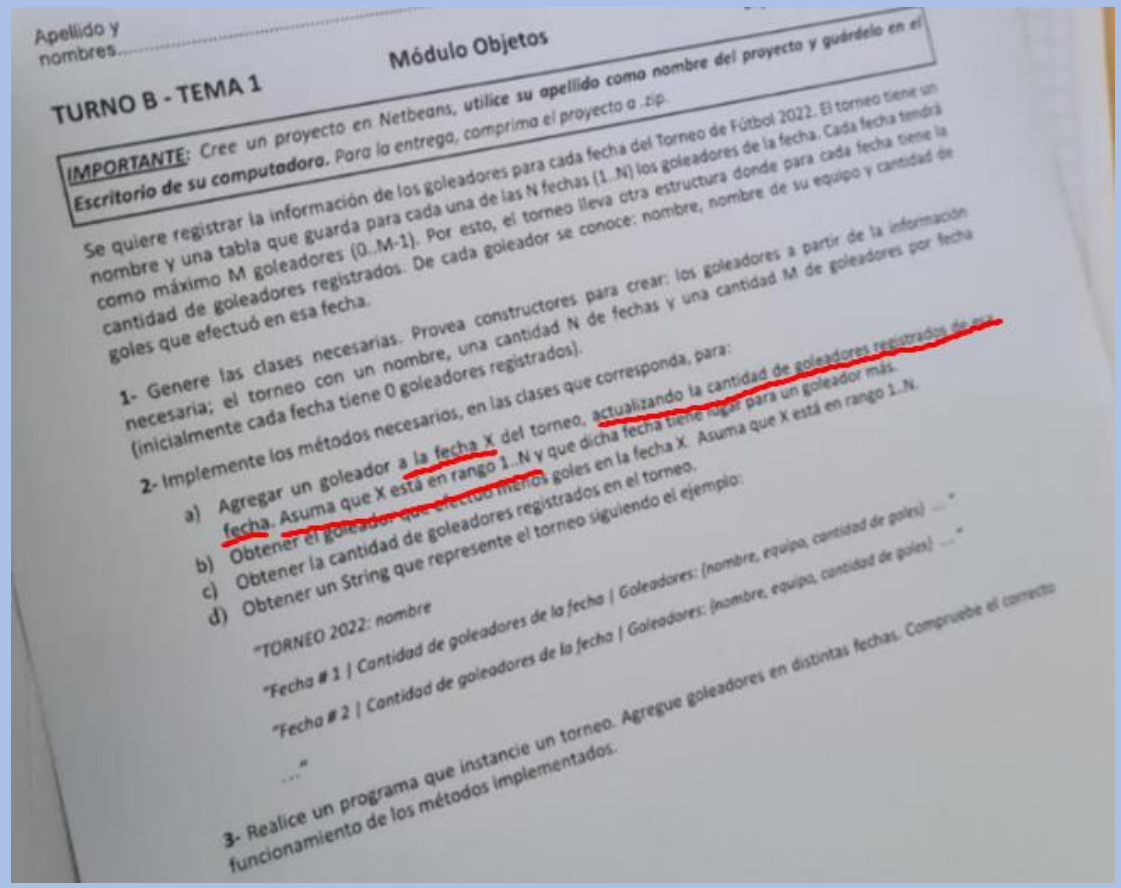
```java
*/
public class Torneo {
    private String nombreTorneo;
    private Goleador tabla[][];

    private int nFechas;
    private int mJugadores;

    private int cantidadGoleadores[]; // dimension Logica de la matriz

    private void inicializarVectorDimls(){
        for (int i =0 ; i < this.nFechas; i++){
            this.cantidadGoleadores[i] = 0;
        }
    }
    //contructor

    public Torneo(String nombreTorneo, int nFechas, int mJugadores) {
        this.nombreTorneo = nombreTorneo;
        this.nFechas = nFechas;
        this.mJugadores = mJugadores;
        this.tabla = new Goleador[mJugadores][nFechas];
        this.cantidadGoleadores = new int[nFechas];
        inicializarVectorDimls();
    }
    public void agregarGoleador(Goleador goleador, int fechaX){
        this.tabla[this.cantidadGoleadores[fechaX]][fechaX] = goleador; //escenario ideal no hago validaciones
        this.cantidadGoleadores[fechaX]++;
    }
    public Goleador jugadorMenosGolesEnFechaX(int fechaX){
        int min = 9999;
        Goleador goleador = null;
        for(int i = 0; i < this.cantidadGoleadores[fechaX]; i++){
            if(this.tabla[i][fechaX].getCantGolesEfectuadosEnEstaFecha() < min){
                min = this.tabla[i][fechaX].getCantGolesEfectuadosEnEstaFecha();
                goleador = this.tabla[i][fechaX];
            }
        }
        return goleador;
    }
    public int cantidadDeJugadoresRegistrados(){
        int suma = 0;
        for (int i = 0; i < this.nFechas; i++){
            suma += this.cantidadGoleadores[i];
        }
        return 0;
    }
```



Apellido y nombres.....

TURNO B - TEMA 1 — Módulo Objetos

IMPORTANTE: Cree un proyecto en Netbeans, **utilice su apellido como nombre del proyecto y guárdelo en el Escritorio de su computadora.** Para la entrega, comprima el proyecto a .zip.

Se quiere registrar la información de los goleadores para cada fecha del Torneo de Fútbol 2022. El torneo tiene un nombre y una tabla que guarda para cada una de las N fechas (1..N) los goleadores de la fecha. Cada fecha tendrá como máximo M goleadores (0..M-1). Por esto, el torneo lleva otra estructura donde para cada fecha tiene la cantidad de goleadores registrados. De cada goleador se conoce: nombre, nombre de su equipo y cantidad de goles que efectuó en esa fecha.

1- Genere las clases necesarias. Provea constructores para crear: los goleadores a partir de la información necesaria; el torneo con un nombre, una cantidad N de fechas y una cantidad M de goleadores por fecha (inicialmente cada fecha tiene 0 goleadores registrados).

2- Implemente los métodos necesarios, en las clases que corresponda, para:

a) Agregar un goleador a la fecha X del torneo, actualizando la cantidad de goleadores registrados de esa fecha. Asuma que X está en rango 1..N y que dicha fecha tiene lugar para un goleador más.
b) Obtener el goleador que efectuó menos goles en la fecha X. Asuma que X está en rango 1..N.
c) Obtener la cantidad de goleadores registrados en el torneo.
d) Obtener un String que represente el torneo siguiendo el ejemplo:

"TORNEO 2022: nombre

"Fecha # 1 | Cantidad de goleadores de la fecha | Goleadores: (nombre, equipo, cantidad de goles) ..."

"Fecha # 2 | Cantidad de goleadores de la fecha | Goleadores: (nombre, equipo, cantidad de goles) ..."

..."

3- Realice un programa que instancie un torneo. Agregue goleadores en distintas fechas. Compruebe el correcto funcionamiento de los métodos implementados.

Matrices

Cargar Filas en orden secucencial

```java
/**
 *
 * @author nahuelArn
 */
public class Padre {

    private String matriz[][];
    private int filaN;
    private int columnaN;

    private int vectorDimls[];
    private int columnaActual;
    private int columnasOcupadas = 0;

    private void inicializarVector() {
        for (int i = 0; i < this.columnaN; i++) {
            this.vectorDimls[i] = 0;
        }
    }

    public Padre(int filaN, int columnaN) {
        this.matriz = new String[filaN][columnaN];
        this.filaN = filaN;
        this.columnaN = columnaN;
        this.vectorDimls = new int[columnaN];
        inicializarVector();
        this.columnaActual = 0;
    }

    public void agregarNombre(String nombre) {
        if (this.columnaActual < this.columnaN) {
            this.matriz[this.vectorDimls[this.columnaActual]][this.columnaActual] = nombre;
            this.vectorDimls[this.columnaActual]++;
            if (this.vectorDimls[this.columnaActual] == this.filaN) {
                if ((this.columnaActual + 1) < this.columnaN) {
                    columnasOcupadas++;
                }
                this.columnaActual++;   //corta con 3 cuando estoy en la ultima columna,pasa a 3 y despues entra en matriz llena
                System.out.println("Valores de columna Actual: " + this.columnaActual);
            }
        } else {
            System.out.println("Matriz llena");
        }
    }

    private void imprimirVector() {
        for (int i = 0; i < this.columnaN; i++) {
            System.out.println("lugar: " + i + " contenido: " + this.vectorDimls[i]);
        }
    }

    public String scrapear() {
        String msj = " ";
        imprimirVector();
        for (int i = 0; i < this.columnasOcupadas + 1; i++) {
            msj += " Columna: " + i;
            for (int j = 0; j < this.vectorDimls[i]; j++) {
                msj += "\n     Fila: " + j + " " + this.matriz[j][i] + " \n";
            }
        }
        return msj;
    }

    @Override
    public String toString() {
        return scrapear();
    }
}
```

```
run:
Valores de columna Actual: 1
Valores de columna Actual: 2
Valores de columna Actual: 3
Matriz llena
lugar: 0 contenido: 4
lugar: 1 contenido: 4
lugar: 2 contenido: 4
 Columna: 0
    Fila: 0 Pepe1
    Fila: 1 Pepe2
    Fila: 2 Pepe3
    Fila: 3 Pepe4
 Columna: 1
    Fila: 0 Pepe4
    Fila: 1 Pepe2
    Fila: 2 Pepe3
    Fila: 3 Pepe4
 Columna: 2
    Fila: 0 Pepe2
    Fila: 1 Pepe2
    Fila: 2 Pepe2
    Fila: 3 Pepe2
```

```java
    public static void main(String[] args) {
        Padre padre = new Padre(4, 3);

        padre.agregarNombre("Pepe1");
        padre.agregarNombre("Pepe2");
        padre.agregarNombre("Pepe3");
        padre.agregarNombre("Pepe4");
        //
        padre.agregarNombre("Pepe4");
        padre.agregarNombre("Pepe2");
        padre.agregarNombre("Pepe3");
        padre.agregarNombre("Pepe4");
//          //
        padre.agregarNombre("Pepe2");
        padre.agregarNombre("Pepe2");
        padre.agregarNombre("Pepe2");
        padre.agregarNombre("Pepe2");

        padre.agregarNombre("Pepe2");

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

2) Implemente los metodos necesarios en las clases que corresponda para
A) ingresar un vehiculo al estacionamiento, teniendo en cuenta que los vehiculos primero completan el primer sector en lugares sucesivos, luego el segundo y asi siguiendo

Matrices

Imagenes mas claras

https://github.com/NahuelArn

Cargar Filas en orden secucencial

```java
public class Padre {
    private String matriz[][];

    private int dimFfilas;
    private int dimFcolumnas;

    private int vectorDimL[];
    private int columnaActual;
    private int columnaCargada;
    private void inicializarVector(){
        for(int i = 0; i < this.dimFcolumnas; i++){
            this.vectorDimL[i] = 0;
        }
    }
    public Padre(int dimFfilas, int dimFcolumnas) {
        this.matriz = new String[dimFfilas][dimFcolumnas];
        this.dimFcolumnas = dimFcolumnas;
        this.dimFfilas = dimFfilas;
        this.vectorDimL = new int[dimFcolumnas];
        inicializarVector();
        this.columnaActual = 0;
        this.columnaCargada = 0;
    }
    public void agregar(String nombre){
         if(this.columnaActual < this.dimFcolumnas) {
            this.matriz[this.vectorDimL[this.columnaActual]][this.columnaActual] = nombre;
            this.vectorDimL[this.columnaActual]++;

            if(this.vectorDimL[this.columnaActual] == this.dimFfilas){
                if(this.columnaActual+1 < this.dimFcolumnas){
                    this.columnaCargada++;
                }
                this.columnaActual++;
            }
        }else{
            System.out.println( x: "Matriz llena");
        }
    }
    public String mostrar(){
        String s = " ";
        for(int i = 0; i < this.columnaCargada+1; i++){
            s+= " Columna: "+ i+ " \n";
            for(int j = 0; j < this.vectorDimL[i]; j++){
                s+= "    Fila: "+ j+" "+this.matriz[j][i] +" \n";
            }
        }
        return s;
    }
```

```java
*/
public class OTraVezOtravez3 {
    public static void main(String[] args) {
        Padre padre = new Padre( dimFfilas: 3, dimFcolumnas: 2);
        padre.agregar( nombre: "Pepe1");
        padre.agregar( nombre: "Pepe2");
        padre.agregar( nombre: "Pepe3");
        padre.agregar( nombre: "Pepe4");
        padre.agregar( nombre: "Pepe5");
        padre.agregar( nombre: "Pepe6");
        padre.agregar( nombre: "Pepe7");
        System.out.println( x: padre.mostrar());
    }

}
```

2) Implemente los metodos necesarios en las clases que corresponda para
A) ingresar un vehiculo al estacionamiento, teniendo en cuenta que los vehiculos primero completan el primer sector en lugares sucesivos, luego el segundo y asi siguiendo

Matrices

Cargar Filas en orden secucencial

```java
public class Padre {

    private Hijo matriz[][];
    private int dimFfilas;
    private int dimFColumnas;

    private int dimLFilasOcupadas;
    private int dimLColumnasOcupadas;

    public Padre(int dimFfilas, int dimFColumnas) {
        this.matriz = new Hijo[dimFfilas][dimFColumnas];
        this.dimFColumnas = dimFColumnas;
        this.dimFfilas = dimFfilas;
    }

    public void agregarHijo(Hijo hijo) { //cargarlo en orden secuencial
        if ((this.dimLColumnasOcupadas < this.dimFColumnas) && (this.dimLFilasOcupadas < this.dimFfilas)) {
            this.matriz[this.dimLFilasOcupadas][this.dimLColumnasOcupadas] = hijo;
            this.dimLFilasOcupadas++;
        }else{
            this.dimLColumnasOcupadas++;
            if(this.dimLColumnasOcupadas < this.dimFColumnas){
                this.dimLFilasOcupadas = 0;
                this.matriz[this.dimLFilasOcupadas][this.dimLColumnasOcupadas] = hijo;
                this.dimLFilasOcupadas++;
            }else{
                System.out.println("\n Matriz completa \n");
            }
        }
    }
    private int testigo(int columnaActual){ //encargado de tirar la dimension de la columna actual
        if(columnaActual < this.dimLColumnasOcupadas){
            return this.dimFfilas;
        }else{
            return this.dimLFilasOcupadas;
        }
    }
    private String scrapear(){
        String s = " ";
        for(int i = 0; i < this.dimLColumnasOcupadas+1; i++){ //+1 porque la columna 0 tambien se cuenta
            s+= "ColumnaActual: "+ i+ " \n";
            for(int j = 0; j < this.testigo(i); j++){
                s+= "   filaActual: "+ j+ " \n";
            }
        }
        return s;
    }
    @Override
    public String toString() {
        return scrapear();
    }

}
```

```java
package pruebaconmatrices;

/**
 *
 * @author nahuelArn
 */
public class Hijo{


}
```

```java
public class PruebaConMatrices {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        Padre padre = new Padre(4,3);
        Hijo hijo = new Hijo();

        //  Primera Columna
        padre.agregarHijo(hijo);
        padre.agregarHijo(hijo);
        padre.agregarHijo(hijo);
        padre.agregarHijo(hijo);
        //  Segunda
        padre.agregarHijo(hijo);
        padre.agregarHijo(hijo);
        padre.agregarHijo(hijo);
        padre.agregarHijo(hijo);
        // TERCERA
        padre.agregarHijo(hijo);
        padre.agregarHijo(hijo);

        //
        System.out.println(padre.toString());
    }

```

```
run:
 ColumnaActual: 0
    filaActual: 0
    filaActual: 1
    filaActual: 2
    filaActual: 3
ColumnaActual: 1
    filaActual: 0
    filaActual: 1
    filaActual: 2
    filaActual: 3
ColumnaActual: 2
    filaActual: 0
    filaActual: 1
```

Esta variante fallba en un Caso

2) Implemente los metodos necesarios en las clases que corresponda para
A) ingresar un vehiculo al estacionamiento, teniendo en cuenta que los vehiculos primero completan el primer sector en lugares sucesivos, luego el segundo y asi siguiendo

Poo

Busqueda en Matriz

Resetear para q salte a la fila

```
public String buscarAuto(String patente){
    boolean sigo = true;
    int i = 0; int j = 0;
    while((i < this.cantPlazas) && (sigo)){
        j = 0;
        while((j < this.cantPlazas) && (sigo)){
            if(lugarDeMatriz[i][j].getPatente().equals( anObject: patente)){
                sigo = false;
            }
            if(sigo)
                j++;
        }if(sigo)
            i++;
    }
    if(!sigo){
        return "El auto de patente: "+patente+ " esta en la plaza: "+j+" piso: "+i;
    }else{
        return "Auto inexistente";
    }
}
```

Matrices

```java
public static void main(String[] args) {
    int filas = 3, columnas = 3;
    int matriz[][] = new int[filas][columnas];
    int i = 0;
    int j = 0;

    System.out.println( x:"Ingrese un numero");
    int numLeido = Lector.leerInt();

    while ((i < filas) && (numLeido != 0)) {
        j = 0;
        while ((j < columnas) && (numLeido != 0)) {
            System.out.println("Se guardo en la fila: " + i + " columna: " + j);
            matriz[i][j] = numLeido;
            System.out.println( x:"Ingrese un numero");
            numLeido = Lector.leerInt();
            j++;
        }
        i++;
    }
```

Cargar Filas en orden secucencial

Matriz 3 * 4

```
0
Valor contenido[1] | Valor contenido[2] | Valor contenido[3] |
Valor contenido[4] | Valor contenido[5] |
```

```java
    //Imprimir hasta donde fue cargado
    int p = 0;
    int pp = 0;
    while (p < filas) {
        pp = 0;
        while ((pp < columnas) && (matriz[p][pp] != 0)) {
            System.out.print("Valor contenido[" + matriz[p][pp]+"]"+ " | ");
            pp++;
        }
        System.out.println( x:"");
        p++;
    }
}
```