# <http://hilite.me/> herramienta para copiar código en html

# Ejercicios JAVA2

## Ejercicio 11: Crear una clase que sea control remoto que tenga un canal, volumen y si está apagado o encendido.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | public class Control\_remoto {  private final static int VOL\_MAX=5;  private final static int NCH\_MAX=10;  private int canal;  private int volumen;  private boolean isEncendido;    Control\_remoto(){  this.canal=0;  this.volumen=0;  this.isEncendido=false;  }    Control\_remoto(int canal,int volumen,boolean isEncendido){  if(canal>NCH\_MAX) {canal=NCH\_MAX;}  if(volumen>VOL\_MAX) {volumen=VOL\_MAX;}  this.canal=canal;  this.volumen=volumen;  this.isEncendido=isEncendido;  } |

Ilustración 1 clase Control remoto

### Crear lo métodos de aumentar de canal, disminuir de canal, subir/bajar volumen y encender o apagar.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55 | public void subirVolumen() {  if(isEncendido) {  if(this.volumen<VOL\_MAX) {  System.out.println("Vol :"+ ++this.volumen);    }else {  System.out.println("Volumen máximo alcanzado: "+this.volumen);  }  }else {  System.out.println("Encienda primero.");  }  }    public void bajarVolumen() {  if(isEncendido) {  if(this.volumen>0) {  System.out.println("Vol :"+ --this.volumen);  }else {  System.out.println("Volumen mìnimo alcanzado: "+ this.volumen);  }  }else {  System.out.println("Encienda primero.");  }  }    public void subirCanal() {  if(isEncendido) {  if(this.canal<NCH\_MAX) {  System.out.println("CH :"+ ++this.canal);  }else {  System.out.println("Canal máximo alcanzado :"+this.canal);  }  }else {  System.out.println("Encienda primero.");  }  }    public void bajarCanal() {  if(isEncendido) {  if(this.canal>0) {  System.out.println("CH :"+ --this.canal);  }else {  System.out.println("Canal mínimo alcanzado "+this.canal);  }    }else {  System.out.println("Encienda primero.");  }  }  public void pulsarBotonEncendido() {  String txt= (isEncendido)?"Apagando...":"Encendiendo..";  isEncendido=!isEncendido;  System.out.println(txt);  } |

Ilustración 2 metodos mando remoto

### Hacer distintas pruebas de subir/bajar volumen, canal o encender/apagar.

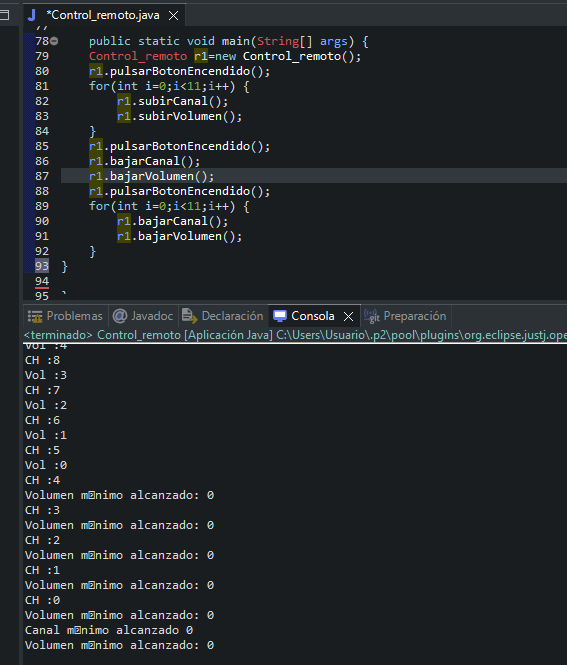


Ilustración 3 pruebas Mando remoto

## Crear una clase coche.

### Con los siguientes atributos: o Marca o Modelo o Velocidad máxima o Tipo combustible o Velocímetro o Tacómetro

public class Coche {

private String marca;

private String modelo;

private int vMax;

private int rvMax;

private String tipocombustible;

private int velocimetro;

private int tacometro;

private boolean isEncendido;

private boolean isMarchaAtras;

private int anguloVolante;

Coche(String marca,String modelo,int vMax,String tipoCombustible)

{

this.marca=marca;

this.modelo=modelo;

this.vMax=vMax;

this.tipocombustible=tipoCombustible;

this.velocimetro=0;

this.tacometro=0;

this.isEncendido=false;

this.isMarchaAtras=false;

this.rvMax=40;

}

### Con los siguientes métodos: o Arrancar o Apagar o Acelerar o Frenar o Girar el volante o Dar marcha atrás ,

public void pulsarBotonStartStop() {

if(this.velocimetro==0) {

this.isEncendido=(this.isEncendido)?false:true;

}else {

System.out.println("Paré el coche primero");

}

}

public void estado() {

System.out.println((this.isEncendido)?"Encendido":"Apagado");

System.out.println("V="+this.velocimetro);

System.out.println("Sentido public void acelerar() {

if(isEncendido) {

if(!isMarchaAtras) {

if(this.velocimetro<=(this.vMax-10)) {

this.velocimetro+=10;

System.out.println("Acelerando 10 km/h.");

}else {

System.out.println("Imposible acelerar la Velocidad actual es :"+this.velocimetro+" igual que la máxima del vehiculo("+this.vMax+")");

}

}else if(isMarchaAtras){

if(this.velocimetro<=(this.rvMax-10)) {

this.velocimetro+=10;

System.out.println("Acelerando 10 km/h.");

}else {

System.out.println("Imposible acelerar la Velocidad actual(reverser) es :" +this.velocimetro);

}

}

}else {

System.out.println("Debe encender el vehiulo primero");

}

};

"+ ((this.isMarchaAtras)?"Reverse":"Directo"));

}

public void darMarchaAtars() {

if(this.velocimetro==0) {

this.isMarchaAtras=true;

}else {

System.out.println("Debe parar el vehiculo primero");

}

};

public void darMarchaAdelante() {

if(this.velocimetro==0) {

this.isMarchaAtras=false;

}else {

System.out.println("Debe parar el vehiculo primero");

}

};

public void frenar() {

if(isEncendido) {

if(!isMarchaAtras) {

if(this.velocimetro>=(0+10)) {

this.velocimetro-=10;

System.out.println("Frenando 10 km/h.");

}else {

System.out.println("Imposible frenar mas la velocidad es: "+this.velocimetro);

}

}else if(isMarchaAtras){

if(this.velocimetro>=(0+10)) {

this.velocimetro-=10;

System.out.println("Frenando 10 km/h.");

}else {

System.out.println("Imposible frenar mas la velocidad es: "+this.velocimetro);

}

}

}else {

System.out.println("Debe encender el vehiulo primero");

}

};

public void girarVolante(int angulo) {

if(angulo<0) {

if((this.anguloVolante+angulo)>=-540) {

this.anguloVolante+=angulo;

System.out.println("girando para la izquierda....");

}else {

this.anguloVolante=-540;

System.out.println("No se pude girar mas para la izquierda");

}

}

if(angulo>0) {

if((this.anguloVolante+angulo)<=+540) {

this.anguloVolante+=angulo;

System.out.println("girando para la derecha....");

}else {

this.anguloVolante=540;

System.out.println("No se pude girar mas para la derecha");

}

}

System.out.println((this.anguloVolante<=0)?"Direccion "+Math.abs(this.anguloVolante)/18+" º a la izquierda":"Direccion "+Math.abs(this.anguloVolante)/18+"º a la derecha");

};

### Hacer pruebas con un coche.

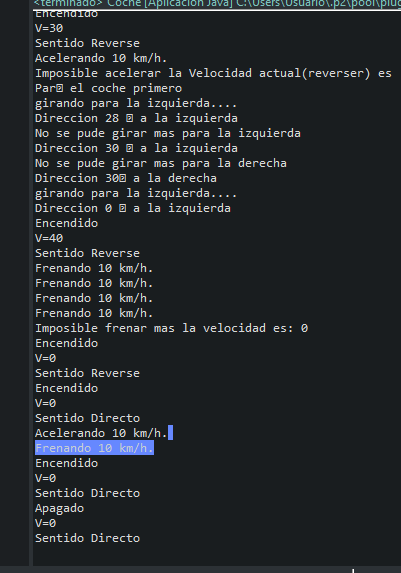
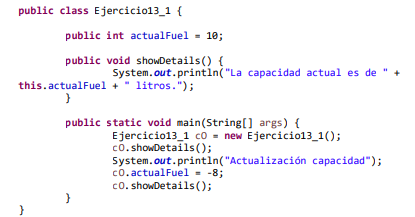


Ilustración 4 resultados coche



Ilustración 5 Sentencias Pruebas

## Ejercicio 13 : A partir de este código:



No permitir introducir valores negativos a actualFuel.

public class Fuel {

private int actualFuel;

Fuel(int actualFuel){

if(actualFuel>=0) {

this.actualFuel=actualFuel;

}else {

this.actualFuel=0;}

}

public void showDetails() {

System.out.println("La capacidad actual es de " +this.actualFuel + " litros.");

}

public void rellenar\_actual\_Fuel(int cantidad) {

if((this.actualFuel+cantidad)>=0) {

this.actualFuel=cantidad;

}else {

this.actualFuel=0;

}

}

public static void main(String[] args) {

Fuel deposito = new Fuel(7);

deposito.showDetails();

Fuel deposito2=new Fuel(-9);

deposito2.showDetails();

//Sacar el nombre de la instancia??? sin poner un atributo nombre dentro de la clase.

deposito.rellenar\_actual\_Fuel(-8);

deposito.showDetails();

deposito.actualFuel=-100;

deposito.showDetails();

}

}

SACAR NOMBRE DE UNA INSTANCIA ¿??????? Sin tenerlo como atributo

## Ejercicio 14: crear clase personas, doctor, policía y profesor y una clase principal donde se llame a los detalles de cada uno de ellos.

import java.time.LocalDate;

public class Persona {

private int dni;

private String nombre;

private String apellido1;

private String apellido2;

private LocalDate fechaNac;

Persona(int dni,String nombre,String apellido1,String apellido2,LocalDate fNac){

this.dni=dni;

this.nombre=nombre;

this.apellido1=apellido1;

this.apellido2=apellido2;

this.fechaNac=fNac;

}

public int getDni() {

return dni;

}

public void setDni(int dni) {

this.dni = dni;

}

public String getNombre() {

return nombre;

}

public void setNombre(String nombre) {

this.nombre = nombre;

}

public String getApellido1() {

return apellido1;

}

public void setApellido1(String apellido1) {

this.apellido1 = apellido1;

}

public String getApellido2() {

return apellido2;

}

public void setApellido2(String apellido2) {

this.apellido2 = apellido2;

}

public String getFechaNac() {

return "Fecha nacimeinto "+fechaNac.toString() +" era el día de la semana "+fechaNac.getDayOfWeek().toString();

}

public void setFechaNac(LocalDate fechaNac) {

this.fechaNac = fechaNac;

}

public String toString() {

String txt="++++++++++++++++++++++++\n";

txt+="Dni:"+this.dni+"\n";

txt+=("Nombre :"+this.nombre+"\n");

txt+=("Apellidos :")+this.apellido1+" "+this.apellido2+"\n";

txt+=(this.getFechaNac());

return txt;

}}

mport java.time.LocalDate;

public class Policia extends Persona {

private String comisaria;

private int distrito;

private final static int NOMINA=2200; Todos lo mismo y cambio en todos ese es el uso?

Policia(int dni, String nombre, String apellido1, String apellido2, LocalDate fNac,String comisaria,int distrito) {

super(dni, nombre, apellido1, apellido2, fNac);

this.comisaria=comisaria;

this.distrito=distrito;

}

public String getComisaria() {

return comisaria;

}

public void setComisaria(String comisaria) {

this.comisaria = comisaria;

}

public int getDistrito() {

return distrito;

}

public void setDistrito(int distrito) {

this.distrito = distrito;

}

public static int getNomina() {

return NOMINA;

}

public String toString() {

String txt="\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

txt+="Dni:"+this.getDni()+"\n";

txt+=("Nombre :"+this.getNombre()+"\n");

txt+=("Apellidos :")+this.getApellido1()+" "+this.getApellido2()+"\n";

txt+=(this.getFechaNac()+"\n");

txt+=("Trabaja en la comisaria :"+this.getComisaria()+"\n");

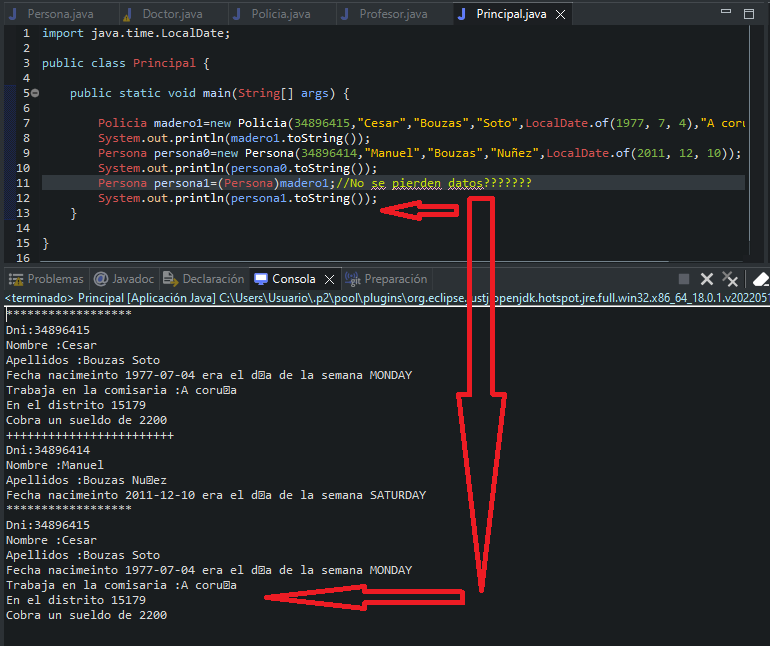
txt+=("En el distrito "+this.getDistrito()+"\n");

txt+=("Cobra un sueldo de "+Policia.getNomina());

return txt;

}

}



Dudas:

* Paso Policia a Persona pero el método sobreescrito me sigue utilizando el de policía ,no se perderían datos??

Es una referencia a un objeto Policia no es propiamente otro objeto.

* Se pude llamar al constructror del padre sin indicar parámetros y que se encomiende él pedirlos o siempre se tienen que crear con paramteros(si los tiene claro).Consructores muy largos.

NO

* Esta bien aplicado? el final static para tener todos los polis la misma nomina y poder cambiársela a todos a la vez.

Entiendo que si si quisiera cambiar el valor en lo cmabiaria ahí para todos

## Ejercicio 15:

### **Crear la clase mercancía con los siguientes atributos : Id ,Nombre , Responsable , Zona , Área ,Estantería Y Cantidad .**

public class Mercancia {

private static String EMPRESA="Cesar.sl";

private int id;

private String nombre;

private String responsable;

private int zona;

private int estanteria;

private int cantidad;

/\*\*

\* **@param** id

\* **@param** nombre

\* **@param** responsable

\* **@param** zona

\* **@param** estanteria

\* **@param** cantidad

\*/

public Mercancia(int id, String nombre, String responsable, int zona, int estanteria, int cantidad) {

this.id = id;

this.nombre = nombre;

this.responsable = responsable;

this.zona = zona;

this.estanteria = estanteria;

this.cantidad = cantidad;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getNombre() {

return nombre;

}

public void setNombre(String nombre) {

this.nombre = nombre;

}

public String getResponsable() {

return responsable;

}

public void setResponsable(String responsable) {

this.responsable = responsable;

}

public int getZona() {

return zona;

}

public void setZona(int zona) {

this.zona = zona;

}

public int getEstanteria() {

return estanteria;

}

public void setEstanteria(int estanteria) {

this.estanteria = estanteria;

}

public int getCantidad() {

return cantidad;

}

public void setCantidad(int cantidad) {

this.cantidad = cantidad;

}

public static String getEMPRESA() {

return EMPRESA;

}

public String toString() {

String txt="";

txt="\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

txt+="Id :"+this.getId()+".\n";

txt+="Nombre: "+this.getNombre()+".\n";

txt+="Responsable : "+this.getResponsable()+".\n";

txt+="Zona : "+this.getZona()+".\n";

txt+="Estanteria : "+this.getEstanteria()+".\n";

txt+="Cantidad :"+this.getCantidad()+".\n";

txt+="\*\*\*\*\*\*\*"+Mercancia.getEMPRESA()+"\*\*\*\*\*\*\*\*\*";

return txt;

}

}

### Crear la clase mercancía perecedera que extienda de mercancía con el atributo fecha caducidad.

**import** **java.time.LocalDate**;

**import** **java.time.LocalTime**;

public **class** **MercanciaPerecedera** extends Mercancia {

private LocalDate caducidad;

private String seccion;

MercanciaPerecedera(int id, String nombre, String responsable, int zona, int estanteria, int cantidad,String seccion) {

super(id, nombre, responsable, zona, estanteria, cantidad);

this.caducidad=LocalDate.now().plusDays(**15**);//fecha de hoy mas **15** dias.

this.seccion=seccion;

}

public LocalDate getCaducidad() {

**return** caducidad;

}

public void setCaducidad(LocalDate caducidad) {

this.caducidad = caducidad;

}

public String getSeccion() {

**return** seccion;

}

public void setSeccion(String seccion) {

this.seccion = seccion;

}

public String toString() {

String txt="";

txt="\*\*\*\*\*\*\*\*\*\*\*\*\*\***\n**";

txt+="Id :"+this.getId()+".**\n**";

txt+="Nombre: "+this.getNombre()+".**\n**";

txt+="Responsable : "+this.getResponsable()+".**\n**";

txt+="Zona : "+this.getZona()+".**\n**";

txt+="Estanteria : "+this.getEstanteria()+".**\n**";

txt+="Cantidad :"+this.getCantidad()+".**\n**";

txt+="Caducidad :"+this.getCaducidad().getDayOfMonth()+

"/"+this.getCaducidad().getMonthValue()+"/"+this.getCaducidad().getYear()+".**\n**";

txt+="Seccion : "+this.getSeccion()+"**\n**";

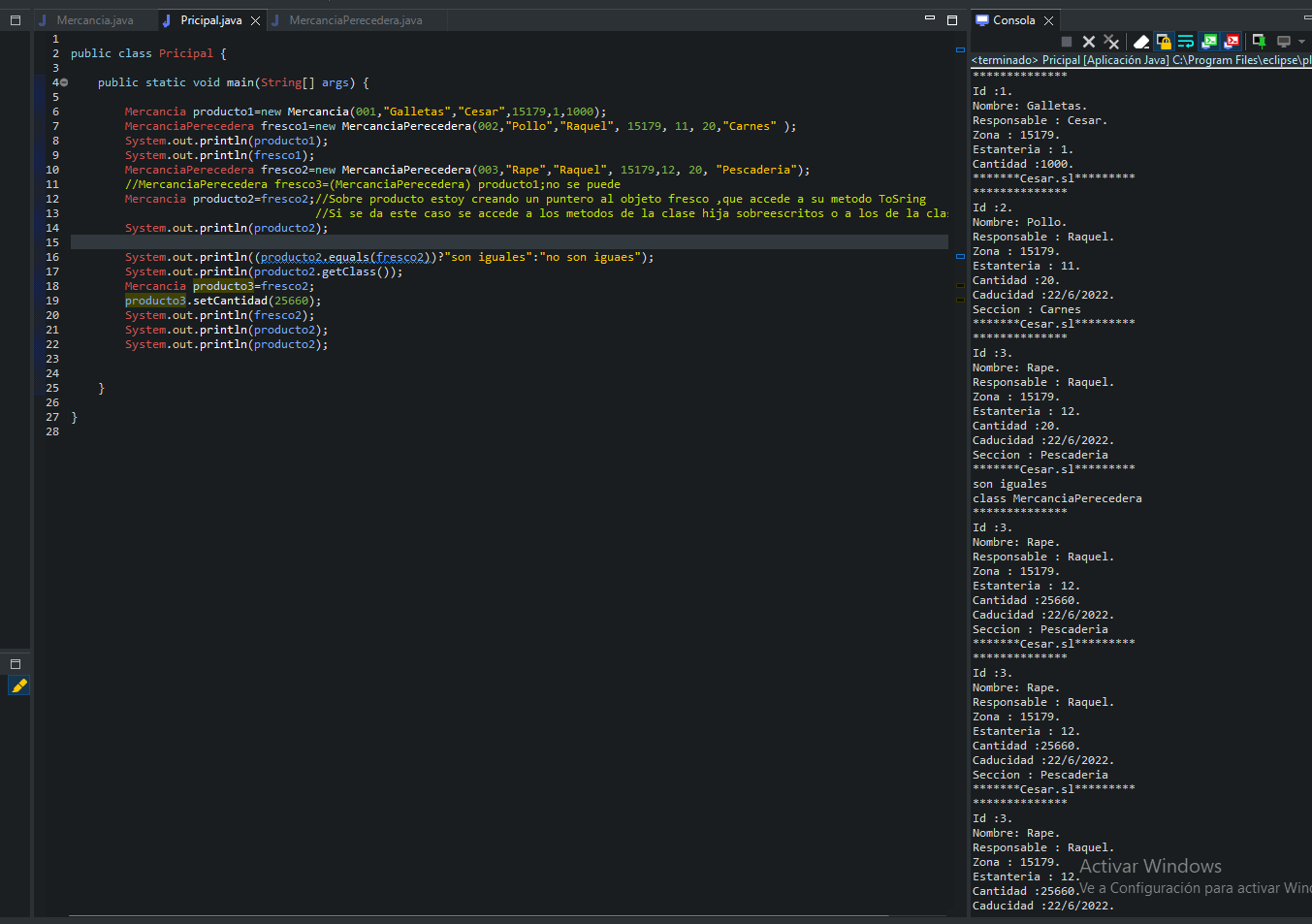
txt+="\*\*\*\*\*\*\*"+Mercancia.getEMPRESA()+"\*\*\*\*\*\*\*\*\*";

**return** txt;

}

}

### Mostrar por pantalla sus características principales.



## Ejercicio 16.

### Crear interfaz máquina (con los métodos encender, apagar y mantenimiento)

public interface Maquina {

public void encender();

public void apagar();

//public void despegar();

public long mantenimiento();

}

### Clases avión que la implemente.

**import** **java.time.LocalDate**;

**import** **java.time.temporal.ChronoUnit**;

public **class** **Avion** implements Maquina{

private LocalDate fechaUltiMante;

private boolean isEncendido;

private boolean isVolando;

Avion(LocalDate fechaUltimoMante,boolean isEncendido, boolean isVolando){

this.fechaUltiMante=fechaUltimoMante;

this.isEncendido=isEncendido;

this.isVolando=isVolando;

}

public void encender() {

**if**(isEncendido) {

System.out.println("Ya está encendido");

}**else** {

this.isEncendido=!this.isEncendido;

}

}

public void apagar() {

**if**(!isEncendido) {

System.out.println("Ya está apagado");

}**else** {

this.isEncendido=!this.isEncendido;

}

}

public long mantenimiento() {

LocalDate ahora=LocalDate.now();

**return** ChronoUnit.DAYS.between(ahora, getFechaMantenimiento());

}

public void despegar() {

**if**(isVolando) {

System.out.println("No puedes depegar ya estas volando");

}**else** {

isVolando=!isVolando;

}

}

public void aterreizar() {

**if**(!isVolando) {

System.out.println("No puedes aterrizar ya estas en el suelo");

}**else** {

isVolando=!isVolando;

}

}

public LocalDate getFechaMantenimiento() {

**return** fechaUltiMante.plusYears(**2**);

}

public String toString() {

String txt="";

txt+="\*\*\*\*\*\*\*\*\*\*"+this.getClass()+"\*\*\*\*\*\*\*\***\n**";

txt+="Motor :"+this.isEncendido+"**\n**";

txt+="Estado :"+this.isVolando+"**\n**";

**if**(mantenimiento()>=**0**) {

txt+="Matenimiento= OK faltan :"+this.mantenimiento()+" dias**\n**";

}**else** {

txt+="Mantenimiento= No Ok! Se pasan:"+this.mantenimiento()+"dias**\n**";

}

**return** txt;

}

}

### Clase tractor que la implemente.

**import** **java.time.LocalDate**;

**import** **java.time.temporal.ChronoUnit**;

**import** **java.time.temporal.Temporal**;

public **class** **Tractor** implements Maquina {

private LocalDate fechaUltimoMantenimeinto;

private boolean isEncendido;

private boolean isArando;

Tractor(LocalDate fechaUltimoManteniemito,boolean isEncendido,boolean isArando){

this.fechaUltimoMantenimeinto=fechaUltimoManteniemito;

this.isEncendido=isEncendido;

this.isArando=isArando;

}

public void encender() {

**if**(isEncendido) {

System.out.println("Ya está encendido");

}**else** {

this.isEncendido=!this.isEncendido;

}

}

public void apagar() {

**if**(!isEncendido) {

System.out.println("Ya está apagado");

}**else** {

this.isEncendido=!this.isEncendido;

}

}

public void arar() {

**if**(isArando) {

System.out.println("Ya está Arando");

}**else** {

this.isEncendido=!this.isEncendido;

}

}

public void dejarArar() {

**if**(!isArando) {

System.out.println("No está arando");

}**else** {

this.isEncendido=!this.isEncendido;

}

}

public long mantenimiento() {

**return** ChronoUnit.DAYS.between(LocalDate.now(), fechaUltimoMantenimeinto.plusYears(**2**));

}

public String toString() {

String txt="";

txt+="\*\*\*\*\*\*\*\*\*\*"+this.getClass()+"\*\*\*\*\*\*\*\***\n**";

txt+="Motor :"+this.isEncendido+"**\n**";

txt+="Estado :"+this.isArando+"**\n**";

**if**(mantenimiento()>=**0**) {

txt+="Matenimiento= OK faltan :"+this.mantenimiento()+"dias**\n**";

}**else** {

txt+="Mantenimiento= No Ok! Se pasan:"+this.mantenimiento()+"dias**\n**";

}

**return** txt;

}

### Main.

**import** **java.time.LocalDate**;

public **class** **principal** {

public static void main (String[] args) {

Maquina maquina0=new Avion(LocalDate.of(**2020**, **5**, **21**) ,false,false);

Avion avion1=new Avion(LocalDate.of(**2020**, **5**, **21**) ,false,false);

Maquina maquina1=avion1;//estoy llamando a un puntero de avion1,

maquina0.apagar();

maquina0.encender();

System.out.println(maquina0);

//maquina1.despegar();;//no funciona y estoy llmando a un puntero de avion1???Solo funciona si Sobrescribo los métodos.

((Avion)maquina1).despegar();

((Avion)maquina1).despegar();

((Avion)maquina1).aterreizar();

maquina1=((Avion)maquina1);

**maquina1.despegar();//No funciona**

}}

Lo que entiendo es que la herencia y la implementación funcionan hacia arriba pero solo con los métodos definidos en la clase padre y si queremos un cambio de comportamiento los reescribimos en el hijo .Si queremos usar métodos en la clase padre lo tenemos que definir en la clase padre .

public interface Maquina {

public void encender();

public void apagar();

//public void despegar();

public long mantenimiento();

}

La clase padre apunta a una estructura hija con un puntero realmente no es un objeto es una referencia al mismo

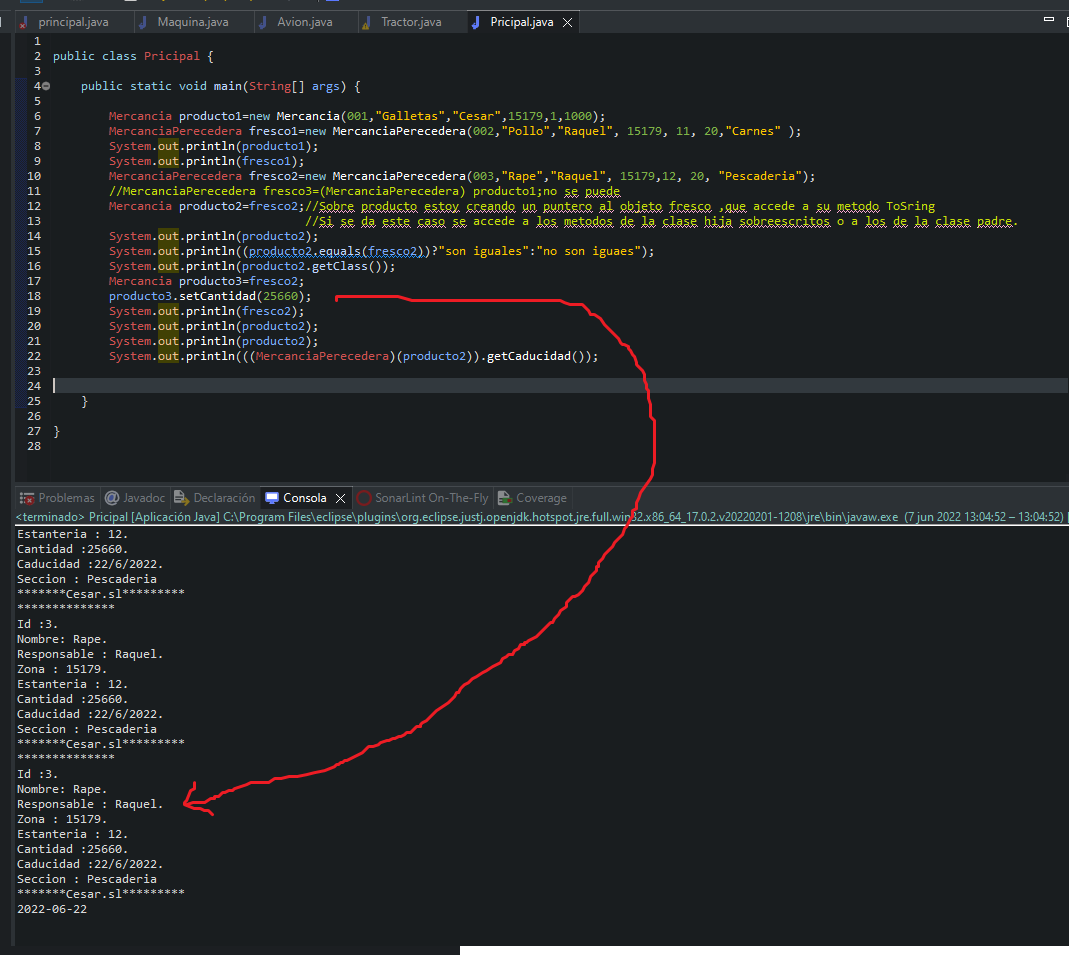


Ilustración 6 cambio un valor y cambia todos los elementos apuntados Ok