

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

2 Ejercicios JAVA2

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

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

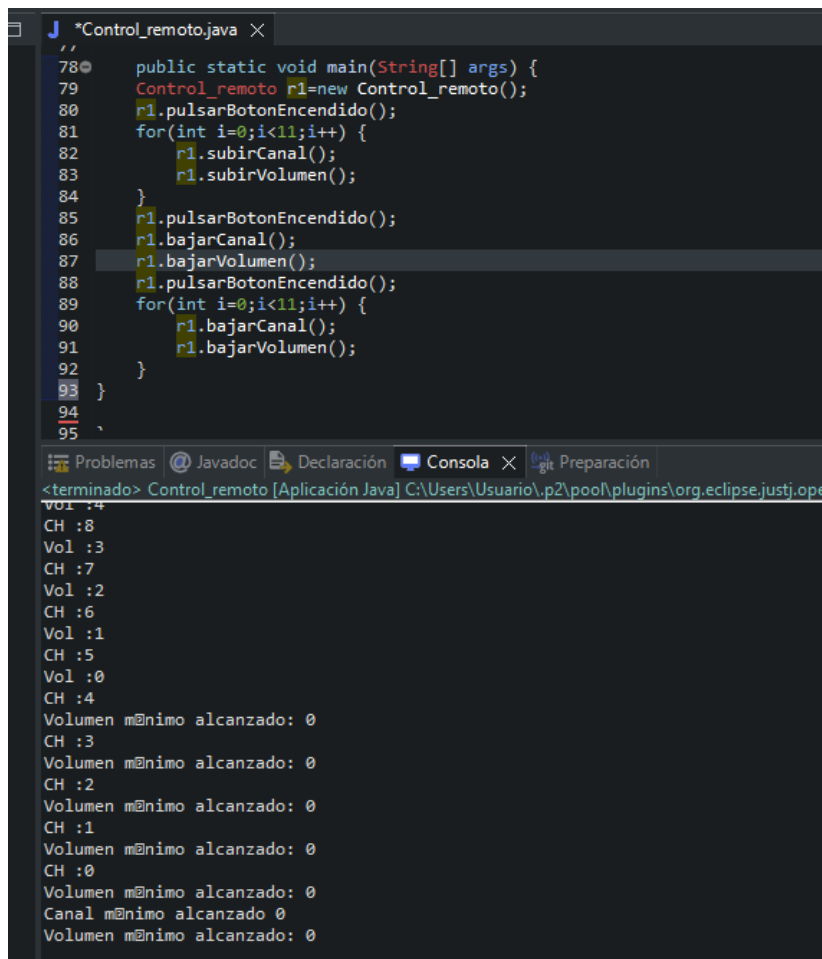
Ilustración 1 clase Control remoto

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

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

Ilustración 2 metodos mando remoto

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



The screenshot shows the Eclipse IDE with a Java file named `*Control_remoto.java` open. The code defines a `main` method that performs a series of remote control actions. The console output shows the execution of these actions, including channel and volume changes, and volume limit warnings.

```
//
78 public static void main(String[] args) {
79     Control_remoto r1=new Control_remoto();
80     r1.pulsarBotonEncendido();
81     for(int i=0;i<11;i++) {
82         r1.subirCanal();
83         r1.subirVolumen();
84     }
85     r1.pulsarBotonEncendido();
86     r1.bajarCanal();
87     r1.bajarVolumen();
88     r1.pulsarBotonEncendido();
89     for(int i=0;i<11;i++) {
90         r1.bajarCanal();
91         r1.bajarVolumen();
92     }
93 }
94
95 `
```

Problemas | Javadoc | Declaración | Consola | Preparación

<terminado> Control_remoto [Aplicación Java] C:\Users\Usuario\.p2\pool\plugins\org.eclipse.justj.open...

```
Vol :4
CH :8
Vol :3
CH :7
Vol :2
CH :6
Vol :1
CH :5
Vol :0
CH :4
Volumen mínimo alcanzado: 0
CH :3
Volumen mínimo alcanzado: 0
CH :2
Volumen mínimo alcanzado: 0
CH :1
Volumen mínimo alcanzado: 0
CH :0
Volumen mínimo alcanzado: 0
Canal mínimo alcanzado 0
Volumen mínimo alcanzado: 0
```

Ilustración 3 pruebas Mando remoto

2.2 Crear una clase coche.

2.2.1 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;  
    }  
}
```

2.2.2 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.        ("Paré el coche primero");
    }
}

public void estado() {
    System.out.        ((this.isEncendido)?"Encendido":"Apagado");
    System.out.        ("V="+this.velocimetro);
    System.out.        ("Sentido public void acelerar() {
    if(isEncendido) {
        if(!isMarchaAtras) {
            if(this.velocimetro<=(this.vMax-10)) {
                this.velocimetro+=10;
                System.out.        ("Acelerando 10 km/h.");
            }else {
                System.out.        ("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.        ("Acelerando 10 km/h.");
            }else {
                System.out.        ("Imposible acelerar la Velocidad
actual(reverser) es :"+this.velocimetro);
            }
        }
    }else {
        System.out.        ("Debe encender el vehiuolo primero");
    }
};

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

public void darMarchaAtars() {
    if(this.velocimetro==0) {
        this.isMarchaAtras=true;
    }else {
        System.out.        ("Debe parar el vehiculo primero");
    }
};

public void darMarchaAdelante() {
    if(this.velocimetro==0) {
        this.isMarchaAtras=false;
```

```

        }else {
            System.out.        ("Debe parar el vehiculo primero");
        }

    };

    public void frenar() {
        if(isEncendido) {
            if(!isMarchaAtras) {
                if(this.velocimetro>=(0+10)) {
                    this.velocimetro-=10;
                    System.out.        ("Frenando 10 km/h.");
                }else {
                    System.out.        ("Imposible frenar mas la velocidad es:
"+this.velocimetro);
                }
            }else if(isMarchaAtras){
                if(this.velocimetro>=(0+10)) {
                    this.velocimetro-=10;
                    System.out.        ("Frenando 10 km/h.");
                }else {
                    System.out.        ("Imposible frenar mas la velocidad es:
"+this.velocimetro);
                }
            }
        }else {
            System.out.        ("Debe encender el vehiulo primero");
        }
    };

    public void girarVolante(int angulo) {
        if(angulo<0) {
            if((this.anguloVolante+angulo)>=-540) {
                this.anguloVolante+=angulo;
                System.out.        ("girando para la izquierda....");
            }else {
                this.anguloVolante=-540;
                System.out.        ("No se pude girar mas para la izquierda");
            }
        }
        if(angulo>0) {
            if((this.anguloVolante+angulo)<=+540) {
                this.anguloVolante+=angulo;
                System.out.        ("girando para la derecha....");
            }else {
                this.anguloVolante=540;
                System.out.        ("No se pude girar mas para la derecha");
            }
        }
        System.out.        ((this.anguloVolante<=0)?"Direccion
"+Math.abs(this.anguloVolante)/18+" º a la izquierda":"Direccion
"+Math.abs(this.anguloVolante)/18+"º a la derecha");
    };

```

2.2.3 Hacer pruebas con un coche.

```
<terminado> Coche [Aplicación Java] C:\Users\Usuario\p2\p001\p10
Encendido
V=30
Sentido Reverse
Acelerando 10 km/h.
Imposible acelerar la Velocidad actual(reverser) es
Par el coche primero
girando para la izquierda....
Direccion 28 a la izquierda
No se pude girar mas para la izquierda
Direccion 30 a la izquierda
No se pude girar mas para la derecha
Direccion 30 a la derecha
girando para la izquierda....
Direccion 0 a la izquierda
Encendido
V=40
Sentido Reverse
Frenando 10 km/h.
Frenando 10 km/h.
Frenando 10 km/h.
Frenando 10 km/h.
Imposible frenar mas la velocidad es: 0
Encendido
V=0
Sentido Reverse
Encendido
V=0
Sentido Directo
Acelerando 10 km/h.
Frenando 10 km/h.
Encendido
V=0
Sentido Directo
Apagado
V=0
Sentido Directo
```

Ilustración 4 resultados coche

```
123
124
125 public static void main(String[] args) {
126     Coche c1=new Coche("Nissan","Xtrail",200,"Diesel");
127     c1.acelerar();
128     c1.pulsarBotonStartStop();
129     c1.estado();
130     c1.acelerar();
131     c1.darMarchaAtars();
132     c1.estado();
133     c1.pulsarBotonStartStop();
134     for(int i=0;i<=19;i++) {
135         c1.acelerar();
136         c1.estado();
137     }
138     c1.pulsarBotonStartStop();
139     c1.frenar();
140     c1.estado();
141     for(int i=0;i<=19;i++) {
142         c1.frenar();
143         c1.estado();
144     }
145     c1.estado();
146     c1.darMarchaAtars();
147     c1.estado();
148     c1.acelerar();
149     c1.acelerar();
150     c1.acelerar();
151     c1.estado();
152     c1.acelerar();
153     c1.acelerar();
154     c1.pulsarBotonStartStop();
155     c1.girarVolante(-510);
156     c1.girarVolante(-200);
157     c1.girarVolante(2000);
158     c1.girarVolante(-541);
159     c1.estado();
```

Ilustración 5 Sentencias Pruebas

2.3 Ejercicio 13 : A partir de este código:

```
public class Ejercicio13_1 {  
  
    public int actualFuel = 10;  
  
    public void showDetails() {  
        System.out.println("La capacidad actual es de " +  
this.actualFuel + " litros.");  
    }  
  
    public static void main(String[] args) {  
        Ejercicio13_1 c0 = new Ejercicio13_1();  
        c0.showDetails();  
        System.out.println("Actualización capacidad");  
        c0.actualFuel = -8;  
        c0.showDetails();  
    }  
}
```

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.        ("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        (7);  
        deposito.        ();  
        Fuel deposito2=new        (-9);  
        deposito2.        ();  
        //Sacar el nombre de la instancia??? sin poner un atributo nombre dentro de la clase.  
        deposito.        (-8);  
        deposito.        ();  
        deposito.actualFuel=-100;  
        deposito.        ();  
    }  
}
```

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

2.4 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;
    }
}
```

```

import 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.                ()+"\n");
        txt+="En el distrito "+this.                ()+"\n");
        txt+="Cobra un sueldo de "+Policia.getNomina());
        return txt;
    }
}

```

```
1 import java.time.LocalDate;
2
3 public class Principal {
4
5     public static void main(String[] args) {
6
7         Policia madero1=new Policia(34896415,"Cesar","Bouzas","Soto",LocalDate.of(1977, 7, 4),"A coru
8         System.out.println(madero1.toString());
9         Persona persona0=new Persona(34896414,"Manuel","Bouzas","Nuñez",LocalDate.of(2011, 12, 10));
10        System.out.println(persona0.toString());
11        Persona personal=(Persona)madero1;//No se pierden datos??????
12        System.out.println(personal.toString());
13    }
14
15 }
16
```

<terminado> Principal [Aplicación Java] C:\Users\Usuario\.p2\pool\plugins\org.eclipse.jdt.launcher\org.eclipse.jdt.launcher.win32.x86_64_18.0.1.v20220517-1800.jar

```
*****
Dni:34896415
Nombre :Cesar
Apellidos :Bouzas Soto
Fecha nacimiento 1977-07-04 era el día de la semana MONDAY
Trabaja en la comisaria :A coruña
En el distrito 15179
Cobra un sueldo de 2200
+++++
Dni:34896414
Nombre :Manuel
Apellidos :Bouzas Nuñez
Fecha nacimiento 2011-12-10 era el día de la semana SATURDAY
*****
Dni:34896415
Nombre :Cesar
Apellidos :Bouzas Soto
Fecha nacimiento 1977-07-04 era el día de la semana MONDAY
Trabaja en la comisaria :A coruña
En el distrito 15179
Cobra un sueldo de 2200
```

Dudas:

- Paso Policia a Persona pero el método sobrescrito 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 constructor del padre sin indicar parámetros y que se encomiende él pedirlos o siempre se tienen que crear con paramteros(si los tiene claro).Constructores 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 cmabiaría ahí para todos

2.5 Ejercicio 15:

2.5.1 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;
    }
}
```

```
}
```

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

```
import java.time.LocalDate;
import java.time.LocalDateTime;
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;
    }

}
```

2.5.3 Mostrar por pantalla sus características principales.

```
1 public class Principal {
2
3
4 public static void main(String[] args) {
5
6     Mercancia producto1=new Mercancia(001,"Galletas","Cesar",15179,1,1000);
7     MercanciaPerecedera fresco1=new MercanciaPerecedera(002,"Pollo","Raquel", 15179, 11, 20,"Carnes" );
8     System.out.println(producto1);
9     System.out.println(fresco1);
10    MercanciaPerecedera fresco2=new MercanciaPerecedera(003,"Rape","Raquel", 15179,12, 20, "Pescaderia");
11    //MercanciaPerecedera fresco3=(MercanciaPerecedera) producto1;no se puede
12    Mercancia producto2=fresco2;//Sobre producto estoy creando un puntero al objeto fresco ,que accede a su metodo ToString
13    //Si se da este caso se accede a los metodos de la clase hija sobrescritos o a los de la clase padre
14    System.out.println(producto2);
15
16    System.out.println((producto2.equals(fresco2))?"son iguales":"no son iguales");
17    System.out.println(producto2.getClass());
18    Mercancia producto3=fresco2;
19    producto3.setCantidad(25660);
20    System.out.println(fresco2);
21    System.out.println(producto2);
22    System.out.println(producto2);
23
24
25 }
26
27 }
28 }
```

```
<terminado> Principal [Aplicación Java] C:\Program Files\ eclipse\
*****
Id :1.
Nombre: Galletas.
Responsable : Cesar.
Zona : 15179.
Estanteria : 1.
Cantidad :1000.
*****Cesar.s1*****
*****
Id :2.
Nombre: Pollo.
Responsable : Raquel.
Zona : 15179.
Estanteria : 11.
Cantidad :20.
Caducidad :22/6/2022.
Seccion : Carnes
*****Cesar.s1*****
*****
Id :3.
Nombre: Rape.
Responsable : Raquel.
Zona : 15179.
Estanteria : 12.
Cantidad :20.
Caducidad :22/6/2022.
Seccion : Pescaderia
*****Cesar.s1*****
*****
son iguales
class MercanciaPerecedera
*****
Id :3.
Nombre: Rape.
Responsable : Raquel.
Zona : 15179.
Estanteria : 12.
Cantidad :25660.
Caducidad :22/6/2022.
Seccion : Pescaderia
*****Cesar.s1*****
*****
Id :3.
Nombre: Rape.
Responsable : Raquel.
Zona : 15179.
Estanteria : 12.
Cantidad :25660.
Caducidad :22/6/2022.
Seccion : Pescaderia
*****Cesar.s1*****
*****
Id :3.
Nombre: Rape.
Responsable : Raquel.
Zona : 15179.
Estanteria : 12.
Cantidad :25660.
Caducidad :22/6/2022.
```

2.6 Ejercicio 16.

2.6.1 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();
}
```

2.6.2 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+="Mantenimiento= OK faltan :"+this.mantenimiento()+" dias\n";
        }else {
            txt+="Mantenimiento= No Ok! Se pasan:"+this.mantenimiento()+"dias\n";
        }
        return txt;
    }
}
```

2.6.3 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 fechaUltimoMantenimeinto, boolean isEncendido, boolean isArando) {
        this.fechaUltimoMantenimeinto = fechaUltimoMantenimeinto;
        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 += "Mantenimiento= OK faltan : " + this.mantenimiento() + "dias\n";
        } else {
            txt += "Mantenimiento= No Ok! Se pasan: " + this.mantenimiento() + "dias\n";
        }
        return txt;
    }
}
```

2.6.4 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 llamando 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

The screenshot shows the Eclipse IDE with a project named 'Principal.java'. The code defines a class hierarchy where 'Principal' is the base class and 'Maquina' is an interface. The 'Principal' class has a 'main' method that creates objects of 'Maquina' and 'Maquina' and prints their details. The console output shows the details of the objects, including their ID, name, responsible person, zone, and maintenance time. A red arrow points from the 'Maquina' object in the code to its corresponding output in the console.

```
1 public class Principal {  
2  
3  
40 public static void main(String[] args) {  
5  
6     Maquina producto1=new Maquina(001,"Galletas","Cesar",15179,1,1000);  
7     MaquinaPerecedera fresco1=new MaquinaPerecedera(002,"Pollo","Raquel", 15179, 11, 20, "Carnes" );  
8     System.out.println(producto1);  
9     System.out.println(fresco1);  
10    MaquinaPerecedera fresco2=new MaquinaPerecedera(003,"Rape","Raquel", 15179,12, 20, "Pescaderia");  
11    //MaquinaPerecedera fresco3=(MaquinaPerecedera) producto1;no se puede  
12    Maquina producto2=fresco2; //Sobre producto estoy creando un puntero al objeto fresco ,que accede a su metodo ToString  
13    //Si se da este error se accede a los metodos de la clase hija sobreescribiendo a los de la clase padre.  
14    System.out.println(producto2);  
15    System.out.println((producto2.equals(fresco2))?"son iguales":"no son iguales");  
16    System.out.println(producto2.getClass());  
17    Maquina producto3=fresco2;  
18    producto3.setCantidad(25660);  
19    System.out.println(fresco2);  
20    System.out.println(producto2);  
21    System.out.println(producto2);  
22    System.out.println(((MaquinaPerecedera) producto2).getCaducidad());  
23  
24 }  
25  
26  
27 }  
28
```

Problemas | Javadoc | Declaración | Console | SonarLint On-The-Fly | Coverage
Terminado: Principal [Aplicación Java] C:\Program Files\ eclipse\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win2.x86_64.j70.2.v2022011208\jre\bin\java.exe (7 jun 2022 13:04:52)
Estanteria : 12.
Cantidad :25660.
Caducidad :22/6/2022.
Seccion : Pescaderia
*****Cesar.s1*****

Id :3.
Nombre: Rape.
Responsable : Raquel.
Zona : 15179.
Estanteria : 12.
Cantidad :25660.
Caducidad :22/6/2022.
Seccion : Pescaderia
*****Cesar.s1*****

Id :3.
Nombre: Rape.
Responsable : Raquel.
Zona : 15179.
Estanteria : 12.
Cantidad :25660.
Caducidad :22/6/2022.
Seccion : Pescaderia
*****Cesar.s1*****
2022-06-22

Ilustración 6 cambio un valor y cambia todos los elementos apuntados