

Prueba identidad + métodos de la clase abstracta ordenador.

Parte 1:

He creado un arrayList para la colección de equipos.

```
public class Partelapp {

    static BufferedReader dato = new BufferedReader(new InputStreamReader(System.in));

    static ArrayList<Ordenador> listaOrdenadores = new ArrayList<>();

    static Sobremesa sobre1 = new Sobremesa("Sobre1", " AMD", 123 , "Asus", "P23", 16, "Intel", 4, "No HDD", false);
    static Sobremesa sobre2 = new Sobremesa("Sobre2", " Nvidia", 143 , "Sony", "Model2", 16, "Intel", 6, "HDD", false);
    static Sobremesa sobre3 = new Sobremesa("Sobre3", " AMD", 173 , "Samsung", "Series 8", 16, "Intel", 4, "SSD", false);

    static Portatil porta1 = new Portatil(13, 300, 111, "porta1", "Asus", 16, "AMD", 4, "SSD", false);
    static Portatil porta2 = new Portatil(13, 300, 222, "porta2", "Apple", 16, "AMD", 4, "HDD", false);
    static Portatil porta3 = new Portatil(13, 300, 333, "porta3", "Acer", 16, "AMD", 4, "NFS", false);
```

El Cuerpo del Main:

```
public static void main(String[] args) {

    listaOrdenadores.add(sobre1);
    listaOrdenadores.add(sobre2);
    listaOrdenadores.add(sobre3);
    listaOrdenadores.add(porta1);
    listaOrdenadores.add(porta2);
    listaOrdenadores.add(porta3);

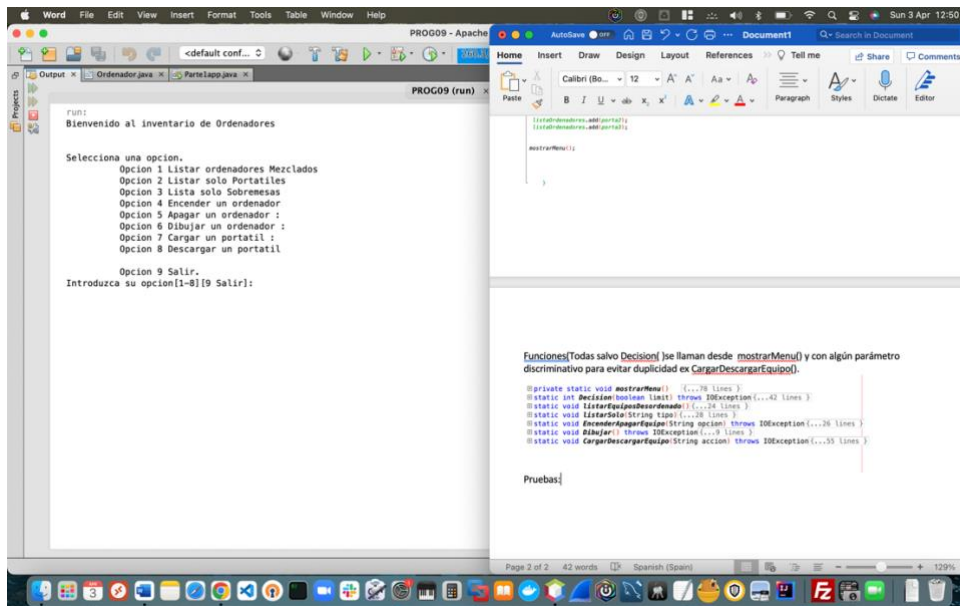
    mostrarMenu();

}
```

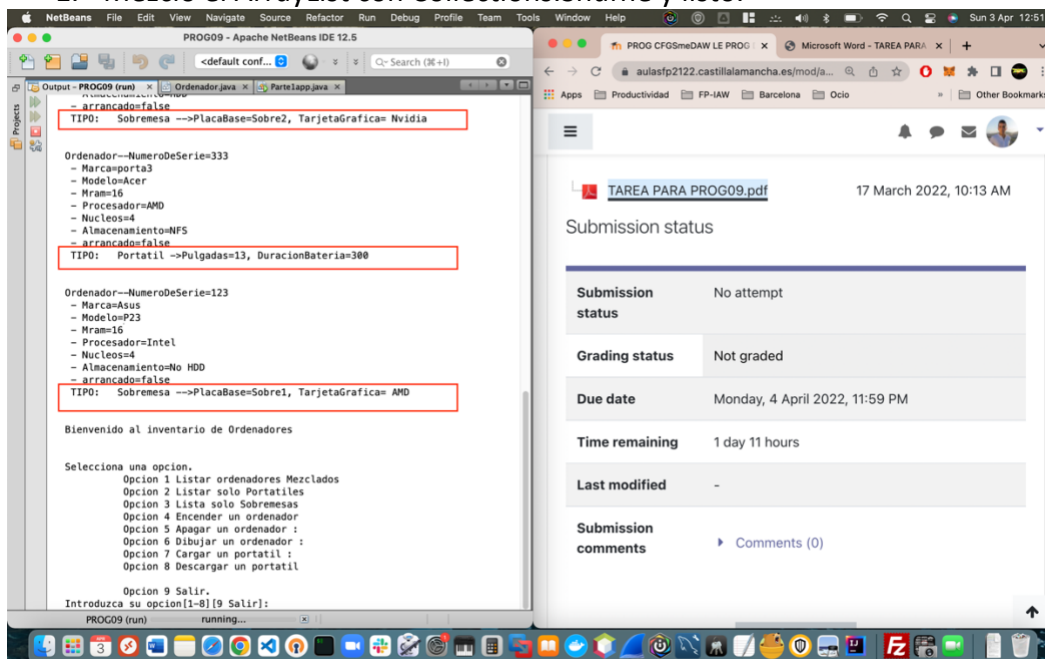
Funciones(Todas salvo Decision() )se llaman desde mostrarMenu() y con algún parámetro discriminativo para evitar duplicidad ex CargarDescargarEquipo().

```
+ private static void mostrarMenu() { ...78 lines }
+ static int Decision(boolean limit) throws IOException { ...42 lines }
+ static void listarEquiposDesordenado() { ...24 lines }
+ static void listarSolo(String tipo) { ...28 lines }
+ static void EncenderApagarEquipo(String opcion) throws IOException { ...26 lines }
+ static void Dibujar() throws IOException { ...9 lines }
+ static void CargarDescargarEquipo(String accion) throws IOException { ...55 lines }
```

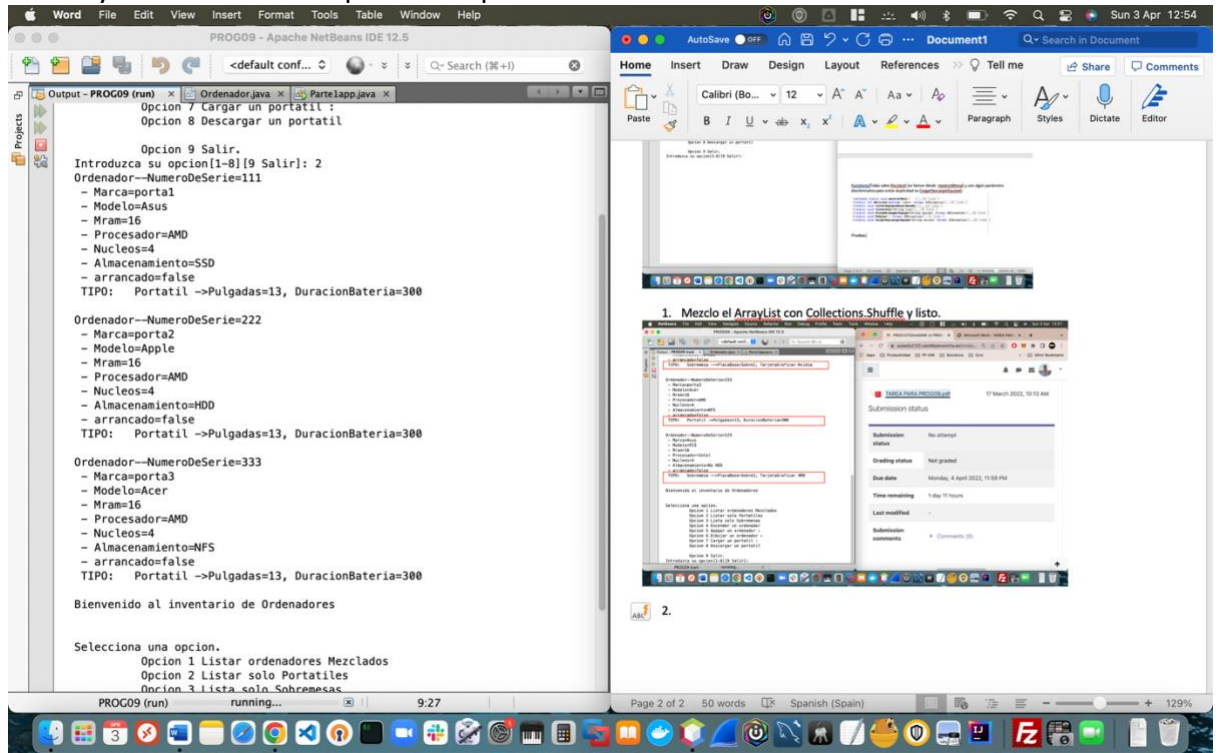
Pruebas:



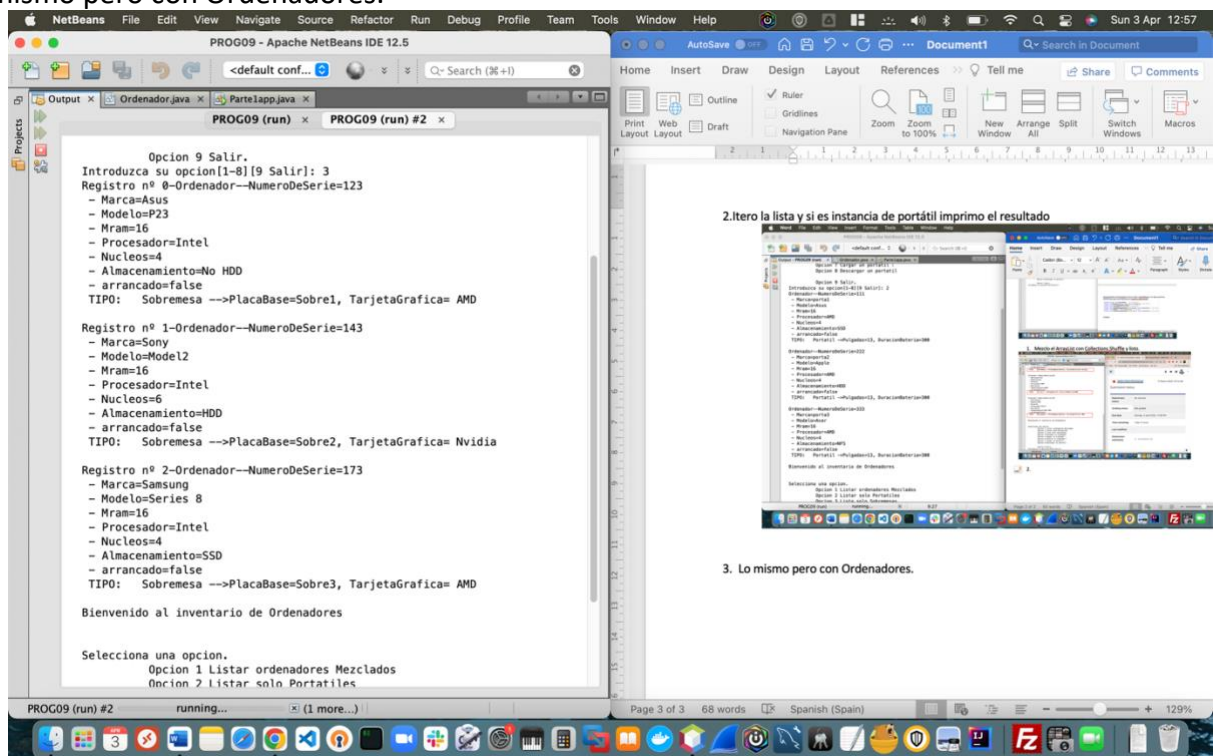
## 1. Mezclo el ArrayList con Collections.Shuffle y listo.



2. Itero la lista y si es instancia de portátil imprimo el resultado

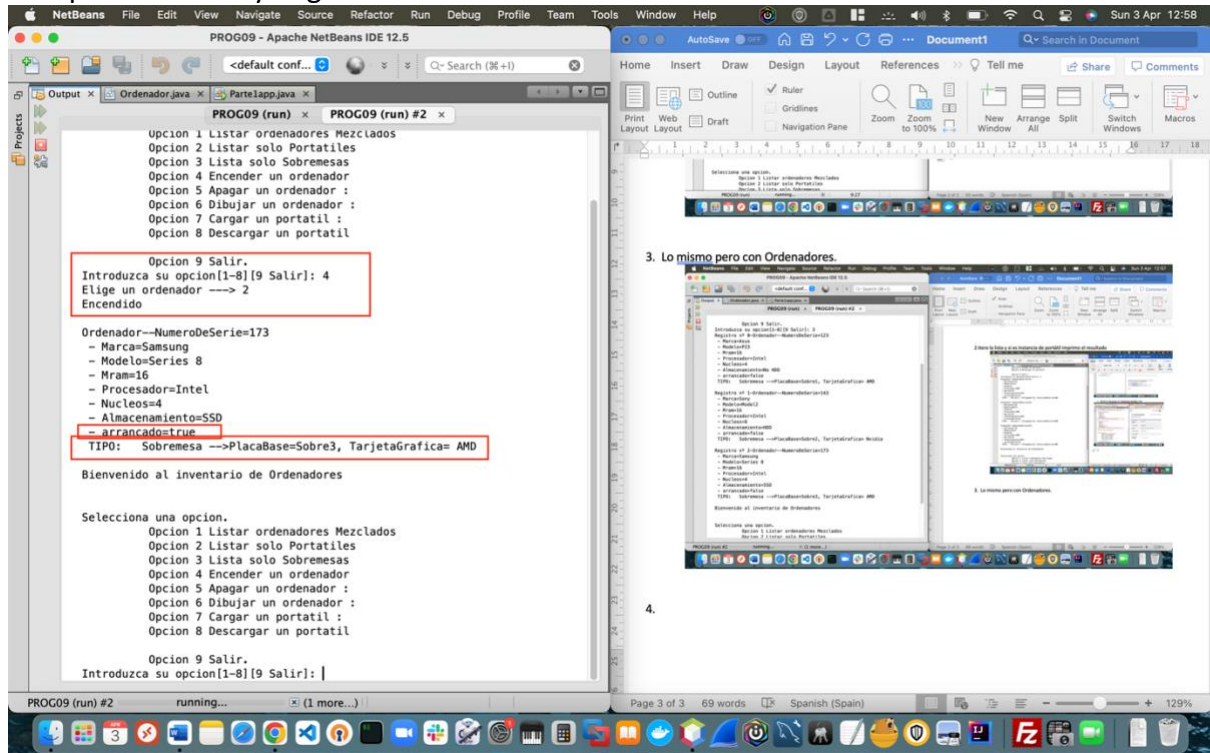


3. Lo mismo pero con Ordenadores.

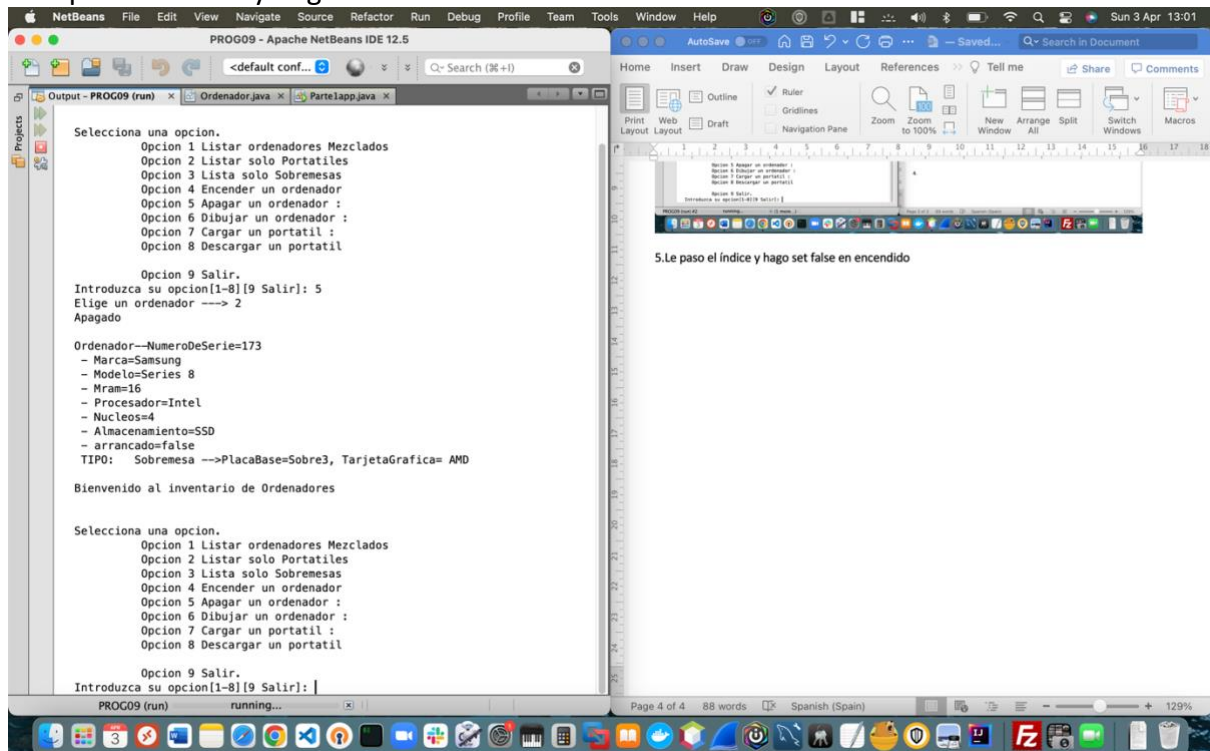




#### 4. Le paso el índice y hago set true en encendido



#### 5. Le paso el índice y hago set false en encendido



[illegible]

NetBeans File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

PROG009 - Apache NetBeans IDE 12.5

default console

Search (Ctrl+F)

Home Insert Draw Design Layout References Tell me Share Comments

Find Layout Layout Draft

Outline

Run

Navigation Pane

Zoom to 100%

New Window All Split Switch Windows Macros

Output - PROG009 (run) - Ordenador.java - C:\Partidos\java -

Opcion 7 Cargar un portatil :

Opcion 8 Descargar un portatil :

Opcion 9 Salir :

Introduzca su opcion(1-9) Salir: 8

Cuantos minutos vas a aladir/guitar ----> 150

Ordenador-NumeroSerie=111

- Marca=portat1
- Modelo=asus
- Mram=16
- Procesador=AMD
- Nucleos=4
- Almacenamiento=550
- arranque=efecto

TIPO: Portatil -->Pulgadas=13, DuracionBateria=350

Ordenador-NumeroSerie=222

- Marca=portat2
- Modelo=apple
- Mram=16
- Procesador=AMD
- Nucleos=4
- Almacenamiento=1000
- arranque=efecto

TIPO: Portatil -->Pulgadas=13, DuracionBateria=350

Ordenador-NumeroSerie=333

- Marca=portat3
- Modelo=lacer
- Mram=16
- Procesador=AMD
- Nucleos=4
- Almacenamiento=1000
- arranque=efecto

TIPO: Portatil -->Pulgadas=13, DuracionBateria=350

Bienvenido al inventario de Ordenadores

Selecciona una opcion.

Opcion 1 Listar ordenadores Mezclados

PROG009 (run)

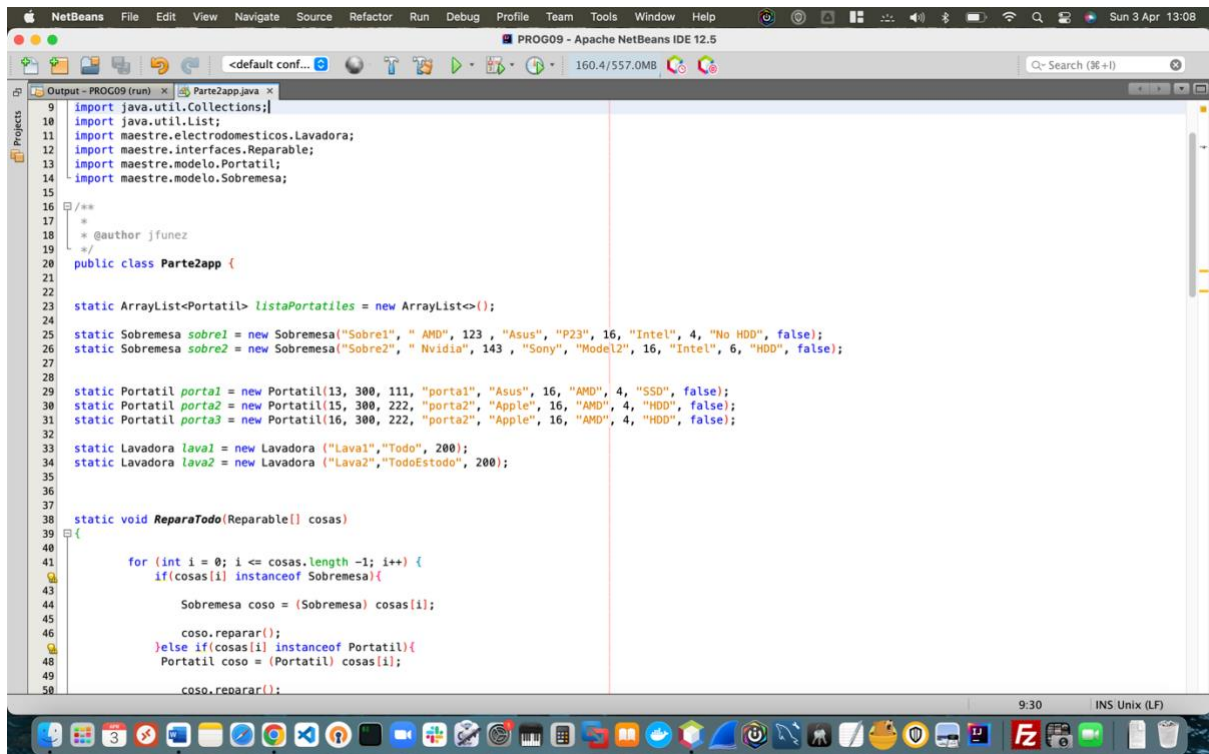
194.6

Page 8 of 8 131 words Spanish (Spain)

129%

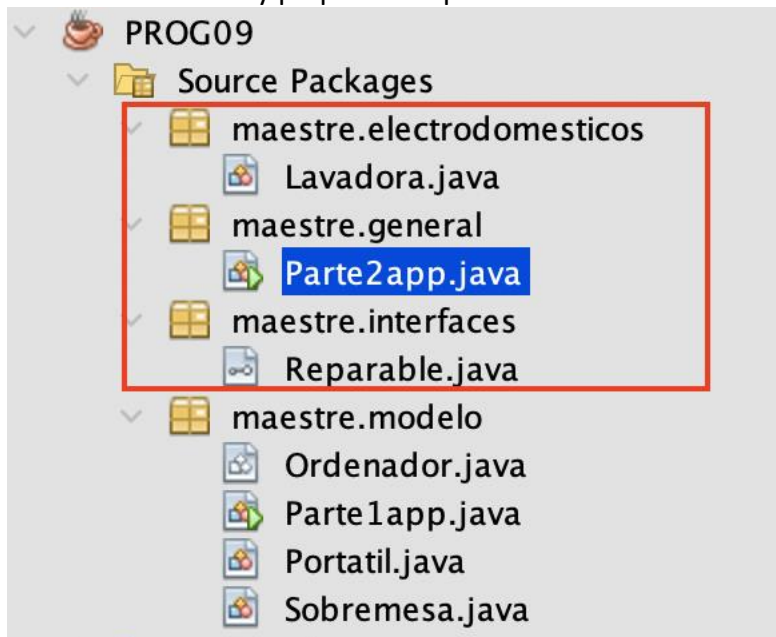
Parte2:

Lista de objetos y metodo principal.



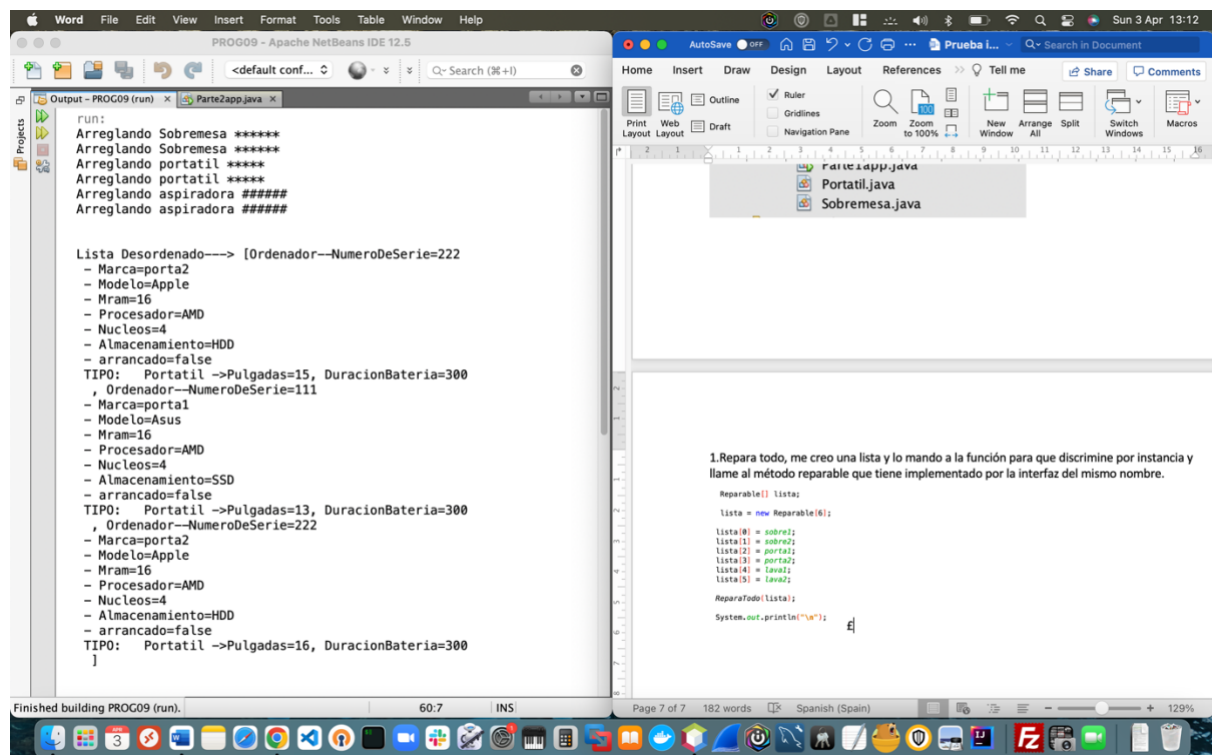
```
9 import java.util.Collections;
10 import java.util.List;
11 import maestre.electrodomesticos.Lavadora;
12 import maestre.interfaces.Reparable;
13 import maestre.modelo.Portatil;
14 import maestre.modelo.Sobremesa;
15
16 /**
17  *
18  * @author jfunez
19  */
20 public class Parte2app {
21
22
23     static ArrayList<Portatil> listaPortatiles = new ArrayList<>();
24
25     static Sobremesa sobremesa1 = new Sobremesa("Sobremesa1", "AMD", 123, "Asus", "P23", 16, "Intel", 4, "No HDD", false);
26     static Sobremesa sobremesa2 = new Sobremesa("Sobremesa2", "Nvidia", 143, "Sony", "Model2", 16, "Intel", 6, "HDD", false);
27
28
29     static Portatil portatil1 = new Portatil(13, 300, 111, "portal", "Asus", 16, "AMD", 4, "SSD", false);
30     static Portatil portatil2 = new Portatil(15, 300, 222, "porta2", "Apple", 16, "AMD", 4, "HDD", false);
31     static Portatil portatil3 = new Portatil(16, 300, 222, "porta2", "Apple", 16, "AMD", 4, "HDD", false);
32
33     static Lavadora lavadora1 = new Lavadora ("Lava1","Todo", 200);
34     static Lavadora lavadora2 = new Lavadora ("Lava2","TodoEstodo", 200);
35
36
37     static void ReparaTodo(Reparable[] cosas)
38     {
39         for (int i = 0; i <= cosas.length -1; i++) {
40             if(cosas[i] instanceof Sobremesa){
41                 Sobremesa coso = (Sobremesa) cosas[i];
42                 coso.reparar();
43             }else if(cosas[i] instanceof Portatil){
44                 Portatil coso = (Portatil) cosas[i];
45                 coso.reparar();
46             }
47         }
48     }
49 }
```

Creacion de Clases y paquetes requeridos



1.Repara todo, me creo una lista y lo mando a la función para que discrimine por instancia y llame al método reparable que tiene implementado por la interfaz del mismo nombre.

```
Reparable[] lista;  
  
lista = new Reparable[6];  
  
lista[0] = sobre1;  
lista[1] = sobre2;  
lista[2] = porta1;  
lista[3] = porta2;  
lista[4] = lava1;  
lista[5] = lava2;  
  
ReparaTodo(lista);  
  
System.out.println("\n");
```



2.Creo el array con los 3 portatiles para desordenarlos transformándolos en colección primero y después los vuelvo a dejar como un array normal.

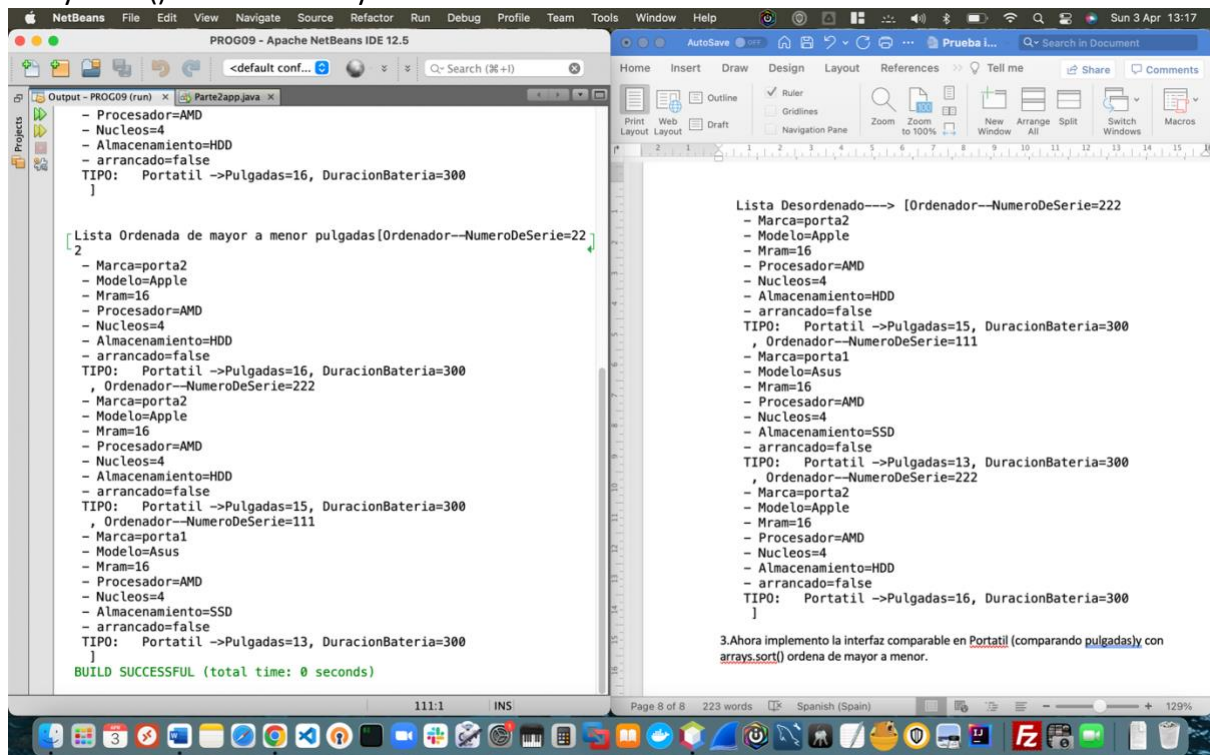
```
Portatil[] listaPortatilesarray;  
  
listaPortatilesarray = new Portatil[3];  
  
listaPortatilesarray[0]= porta1;  
listaPortatilesarray[1]= porta2;  
listaPortatilesarray[2]= porta3;
```



```
Lista Desordenado---> [Ordenador--NumeroDeSerie=222
- Marca=porta2
- Modelo=Apple
- Mram=16
- Procesador=AMD
- Nucleos=4
- Almacenamiento=HDD
- arrancado=false
TIP0:  Portatil ->Pulgadas=15, DuracionBateria=300
, Ordenador--NumeroDeSerie=111
- Marca=porta1
- Modelo=Asus
- Mram=16
- Procesador=AMD
- Nucleos=4
- Almacenamiento=SSD
- arrancado=false
TIP0:  Portatil ->Pulgadas=13, DuracionBateria=300
, Ordenador--NumeroDeSerie=222
- Marca=porta2
- Modelo=Apple
- Mram=16
- Procesador=AMD
- Nucleos=4
- Almacenamiento=HDD
- arrancado=false
TIP0:  Portatil ->Pulgadas=16, DuracionBateria=300
]
```



3. Ahora implemento la interfaz comparable en Portatil (comparando pulgadas) y con `arrays.sort()` ordena de mayor a menor.



Eso es todo.

Todo el código que puede resultar confuso está comentado, he practicado tanto con `ArrayList`, `List` como con `Array` y colecciones para fusionar ambas funcionalidades.