Project

Scenario - OraclProduction

OraclProduction Ltd are specialists in producing production line manufacturing plants.

They could be asked to create a production plant for any type of product ranging from a simple packaging system to a variety of electronic devices.

Recently they have been asked to create a production line for multimedia devices which include music and movie players. They wish to employee you to design a template in Java for creating and recording all future production line items. For this particular production facility you will only implement a concrete class for music and movie players.

Your task is to create a flexible structure that could be used in any production line. This structure would then allow easy modification to handle different products.

Step 1

Create an interface called **Item** that will force all classes to implement the following functions.

- A constant called manufacturer that would be set to "OracleProduction".
- A method setProductionNumber that would have one integer parameter
- A method setName that would have one String parameter
- A method getName that would return a String
- A method getManufactureDate that would return a Date
- A method getSerialNumber that would return an int

Step 2

All items will have a pre-set type. Currently there are 4 types. Create an enum called **ItemType** that will store the following information.

Type	Code
Audio	AU
Visual	VI
AudioMobile	AM
VisualMobile	VM

Step 3

Create an abstract type called **Product** that will implement the Item interface. Product will implement the basic functionality that all items on a production line should have. Add the following fields to Product

- int serialNumber
- String manufacturer
- Date manufacuredOn
- String name

Add an integer class variable called currentProductionNumber. This will store the next number to be assigned to serialNumber.

Complete the methods from the interface Item.

Add a constructor that will take in the name of the product and set this to the field variable name. You will also assign a serial number from the currentProductionNumber. The currentProductionNumber should be incremented in readiness for the next instance.
Set manufacturedOn as the current date and time.

Add a toString method that will return the following: (example data shown).

```
Manufacturer  : OraclProduction
Serial Number : 1
Date          : Thu May 14 15:18:43 BST 2015
Name          : Product Name
```

Step 4

All of the items on this production line will have basic media controls. Create an interface called MultimediaControl that will define the following methods.

- public void play();
- public void stop();
- public void previous();
- public void next();

Step 5

We require a concrete class that will allow us to capture the details of an audio player. Create a class called AudioPlayer that is a subclass of Product and implements the MultimediaControl interface.

The class will have 2 fields

- String audioSpecification
- ItemType mediaType

Create a constructor that will take in 2 parameters – name and audioSpecification.
The constructor should call its parents constructor and also setup the media type.

Implement the methods from the MultimediaControl interface by simply writing the action to the console. E.g. in play System.out.println("Playing"); Normally we would have code that would instruct the media player to play, but we will simply display a message.
Create a toString method that will display the superclasses toString method, but also add rows for Audio Spec and Type.

Step 6

Create a driver class for AudioPlayer that will test to see whether we can instantiate occurrences of it, use the media controls and print out their details to the console.

Step 7

The production facility will also create portable movie players. The main difference between these and the audio players is that they contain screens.

Create an enum called MonitorType that will store

Type
LCD
LED

Step 8

Create an interface called ScreenSpec. This will define 3 methods:

- public String getResolution();
- public int getRefreshRate();
- public int getResponseTime();

Step 9

Create a class called Screen that implements ScreenSpec. Add three fields
String resolution
int refreshrate
int responsetime

Complete the methods from the ScreenSpec interface.

Add a toString method that will return the details of the 3 field in the same format as the Product Class.

Step 10

Create a Driver class for Screen that tests the functionality of the screen class

Step 11

Create a class called MoviePlayer that extends Product and implements MultimediaControl.

Add 2 fields to this class called screen and monitor type and assign appropriate types to them.

Complete the methods from the MultimediaControl interface in a similar fashion to the audio player.

Create a toString method that calls the product toString, displays the monitor and the screen details.

Step 12

Create a diver class to test the functionality of the movie player.

Step 13

The audio players and the movie players share the same control interface on the physical devices. The control interface does not care if the device is a video player or an audio player. Create a driver class that will demonstrate that any class that implements the MultimediaControl Interface would be able to be instantiated and use its methods used no matter if it was an audio or movie player.

Step 14

Add functionality to your classes that would allow them to be sorted by name with the Collections.sort method.

Step 15

Create a method called print that would take your collection and list all of the contents. It should handle all of your classes. You should also demonstrate the Collections.sort method.

Step 16

Create a method called print that would take your collection and list all of the contents. It should handle all of your classes

Step 17 – Bonus

This step is optional and challenging.