Problem description

You need to write an application to manage information of products that are sold per item (item-based products) as well as products that are sold by weight (weight-based products). Each product has a barcode of 12 digits.

The barcodes of products that are sold per item contain information on the manufaturer of the product as well as the name of the product. The barcodes of these products have a 1 as first digit. These products have a name, a manufacturer and a unit price (in cents).

The barcodes of items that are sold by weight contain information on the weight of the product as well as the name of the product. The barcodes of these products have a 2 as first digit. These products have a name, a unit price (in cents per kilogram) as well as a weight (in gram).

For each item-based product, the cost is the unit price plus tax. On the other hand, the cost of a weight-based product is the weight of the product multiplied with its unit price, plus tax. A tax rate of 15% is used for both types of products.

Example data

Item-based products			
Barcode	**Name of product**	**Manufacturer**	**Unit price**
178030640227	Milk, 1L	Clover	1550
167832345047	Tuna	I&J	1500
145672345751	Oros Orange Squash	Brookes	3400

Weight-based products			
Barcode	**Name of product**	**Unit price**	**Weight**
245134867532	Bananas	4300	540
251336424037	Apples	3350	135
245134342471	Bananas	4300	450
267897542569	Rump steak	12500	250

This sample data is given in a single text file (`productdata.txt`). In the text file, data items for each product are stored in the order given in the two tables above and are separated by hash (#) characters. Note that a different text file will be used for the marking of the exams! The contents of `productdata.txt` is as follows:

```
267897542569#Rump steak#12500#250
245134342471#Bananas#4300#450
178030640227#Milk, 1L#Clover#1550
167832345047#Tuna#I&J#1500
251336424037#Apples#3350#135
145672345751#Oros Orange Squash#Brookes#3400
245134867532#Bananas#4300#540
```

You are also provided with the interface **Taxable.java** as well as the following classes: **Product.java**, **ItemBasedProduct.java** and **WeightBasedProduct.java**

QUESTION 1: **Application**

Based on the given information, create an application in which you do the following:

Create a single array that must be able to contain both **ItemBasedProduct** and **WeightBasedProduct** objects. Provide for a maximum of 25 objects. Read the data from the given `productdata.txt` file to create **ItemBasedProduct** and **WeightBasedProduct** objects and store them in the created array. Take note that you have to read the data from the given file. Also take note that both types of objects are to be stored in a single array. Apply exception handling. (15)

Sorting

Write code to sort the array of products:

- The first digit of the barcode indicates the type of product: Item-based products start with 1 and weight-based products start with 2.
- Now use **Arrays.sort()** to sort the array of products and then display a list of records of the sorted products. The products must be sorted alphabetically on their names, HOWEVER, all the item-based products must be at the start of the list and all the weight-based products must be at the end of the list as shown in the example output given below.
- Add the necessary code to the product classes to facilitate the sorting.
- HINT: Use a temporary sorting field.

Do not use the **toString** method to display any data. The output must be in the same format as shown in the example output. (15)

Sort output

```
List of products
Barcode 178030640227 is Milk, 1L from Clover, unit price 1550c
Barcode 145672345751 is Oros Orange Squash from Brookes, unit price 3400c
Barcode 167832345047 is Tuna from I&J, unit price 1500c
Barcode 251336424037 is Apples, unit price 3350c/Kg, weight 135 grams
Barcode 245134342471 is Bananas, unit price 4300c/Kg, weight 450 grams
Barcode 245134867532 is Bananas, unit price 4300c/Kg, weight 540 grams
Barcode 267897542569 is Rump steak, unit price 12500c/Kg, weight 250 grams
```

Serialization

Write code using the **java.io.Serializable** interface to serialize each of the objects in the array to a single file called `products.ser`. Take note that you have to serialize each object by itself and not serialize the array itself.

Write a separate test application class to read the product objects from the file `products.ser` and display the barcode, name and cost of each object on the screen as is shown in the example output below. Note that the values of all the instance variables must be read from the file. You do not need to save the products in an array, you may display them as they are read.

Furthermore, the total cost of al the products in the array as well as the total weight of all the weight-based products must be calculated and displayed as shown in the example output.

Make sure that your programs can handle all possible exceptions! (10)

Serialization output

```
Cost of each product
Barcode:  267897542569:  Rump steak:  R35.93
Barcode:  245134342471:  Bananas:  R22.25
Barcode:  178030640227:  Milk, 1L: R17.82
Barcode:  167832345047:  Tuna:  R17.25
Barcode:  251336424037:  Apples:  R5.19
Barcode:  145672345751:  Oros Orange Squash:  R39.10
Barcode:  245134867532:  Bananas:  R26.70
End of file reached.

Total cost of products:  R164.24

Total weight of weight-based products:  1375 grams
```