Задание

Создать приложение, способное читать и писать в csv-file (<https://ru.wikipedia.org/wiki/CSV> )

1.

Написать класс, хранящий данные о товаре(Product) ( цена, название, артикул, дата изготовление, годность… можно добавить свое по желанию)

Класс должен реализовать интерфейс Comparable

Написать класс, реализующий интерфес

Interface CSVReader{

List<Product> readFromFile(Strig fileName);

}

**public static void** writeToFile(String fileName, List<Product> newData) {  
 File file = **new** File(fileName);  
 **if** (!file.exists()) {  
 **try** {  
 file.createNewFile();  
 } **catch** (IOException e) {  
 ***//do smth***  
 }  
 }  
 PrintWriter out = **null**;  
 **try** {  
 out = **new** PrintWriter(file);out.print(*listToCSV*(newData));  
 ***//do smth***  
  
 } **catch** (FileNotFoundException e) {  
 ***//do smth***  
 } **finally** {  
 *closeStream*(out);  
 }  
}

**private static void** closeStream(Closeable stream) {  
 **if** (stream != **null**) {  
 **try** {  
 stream.close();  
 } **catch** (IOException e) {  
 //***LOGGER***.error(String.*format*(**"Can't close %s"**, stream));  
 }  
 }  
}

Написать класс реализующий интерфейс CSVWriter{

void writeProductListToSCV(Strig destinationFileName, List<Product> newData, boolean appendToFile);

void writeToSCVRandomData(Strig destinationFileName, int lineValue);

}

Если append – true, необходимо дописать новый список в конец файла

В противном случае затереть имеющиеся в нем данные, если файл существует.

1.2 Написать метод вызывающий оба класса и провверяющий, что все считано и записано верно.

**public static void** generateAndWriteMatrix(String fileName, List<Product> newData, boolean appendToFile) {  
 File file = **new** File(fileName);  
 **if** (!file.exists()) {  
 **try** {  
 file.createNewFile();  
 } **catch** (IOException e) {  
 //***LOGGER***.error(String.*format*(**"Can't create file %s"**, fileName));  
 }  
 }  
 PrintWriter out = **null**;  
 **try** {  
 out = **new** PrintWriter(file);

for (Product p : newData){  
 out.print(*rowToCSV*(*p));*   
 }  
 //***LOGGER***.info(String.*format*(**"File %s was written"**, fileName));  
  
 } **catch** (FileNotFoundException e) {  
 //***LOGGER***.error(String.*format*(**"File %s wasn't created"**, fileName));  
 } **finally** {  
 *closeStream*(out);  
 }  
}

2. ( продолжение)

Создать класс, вызывающий Reader,

Класс должен реализовывать интерфейс:

Interface CSVHolder {

boolean writeToCSVNewData(Strig destinationFileName, List<Product> newData);

boolean addNewDataAndSortBy (Strig destinationFileName, List<Product> newData, CompareType typeToSortList);

}

Написать енум, хранящий способы сортировки продуктов

Enum CompareType(

BY\_PRICE, BY\_NAME, BY\_FRESNESS, DEFAULT

}

В этом же классе создать 3 компаратора(переменные static final, которые будут являться анонимными классами) и переопределить у них метод compare для разных полей класса Product.

В методе addNewDataAndSortBy будет осуществляться проверка на то, какой из способов сортировки пришел и соответственно выбираться компаратор.

3. Написать 2 своих класса наследованные от Exception( можно выбрать более точный наследник ),

Один валится, если на позицию введено неверное значение ( например вместо int была введена строка),

Другой если в csv-файле было написано неверное количество строк ( больше или меньше в любой строке)

4.

Выполнение данного задания предусматривает предварительное освоение maven

Добавить логгирование

В pom.xml будет приблизительно такая запись

*<?***xml version="1.0" encoding="UTF-8"***?>*<**project xmlns="http://maven.apache.org/POM/4.0.0"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"**>  
 <**modelVersion**>4.0.0</**modelVersion**>  
  
 <**groupId**>CSVCreator</**groupId**>  
 <**artifactId**>CSVCreator</**artifactId**>  
 <**version**>1.0-SNAPSHOT</**version**>  
  
 <**packaging**>jar</**packaging**>  
  
 <**dependencies**>  
 <**dependency**>  
 <**groupId**>log4j</**groupId**>  
 <**artifactId**>log4j</**artifactId**>  
 <**version**>1.2.17</**version**>  
 </**dependency**>  
 </**dependencies**>  
   
</**project**>

5. Создать класс матрица(**Matrix**), хранящий матрицу данных

У матрицы есть List<String> header – список шапки нашего CSV- файла

Каждый элемент header имеет вид ColumnName(Type) , например Article(String) и Price (Integer)

У матрицы есть

**private** List<Row> **matrix**;

**public class** Row {  
  
 **private** List<Comparable> **members**;  
 **private** String **stringValueOfRow**;

}

Мы заранее не знаем, сколько столбцов будет в scv и какими будут их типы, но знаем, что они будут из списка наших заданных типов (Int, String) для начала.

Сделать сортировку элементов по любому столюбцу класса **Matrix**

6. Выполнение дынного задание предусматривает ознакомление с многопоточностью.

Сделать возможность сортировки файла размером более 1гб.

В классе-райтере, добавить возможность создавать файл нужного размера.