Software Design and Implementation Seminar E

This exercise involves the design and implementation of a Simple Factory class to allow the creation of different materials designed for Seminar D. The outcome of this exercise will be logged with and signed off by your lab tutor by **Friday 11th of March**.

Exercise

As part of the development team for TrekStar pictures, you have already been involved in the design of classes that will handle the project data in the system. Apart from designing those classes you were required to code functionality to:

1. Read the details of a collection of materials related to a project from a text file (e.g. csv file) and create an object for each material.
2. Add new materials to a database, persistently (added to the file!).

Remember that data model classes shouldn't really be aware of where they will be stored, so be careful where you implement that functionality!

So far, you are not entirely sure as how this data model will fit in the complete system. However, from the 2 previous implementation tasks (load materials from file, add new ones and store to a file) you have realised that the implementation is too coupled with the concrete classes (DVD, VHS, Blu-ray, etc.). Your code probably looks something similar (**but not necessarily exactly**) to the following:

```
if (type == "Blu-ray")

{

      material = new Bluray(some parameters will go here);

}

else if (type == "VHS"){

      material = new VHS(some parameters will go here);

}

else if….

myCollection.insert(material);
```

This code might be currently in your main function and at this point that is enough, as you are unaware of how the rest of the system will look like. The problem here is that this portion of code might be repeated for creation of new materials by the user or by reading them from a file, and from other possible functionalities of the system. You have decided then that it is better to encapsulate that creation code in a Simple Factory class so that in the future you know exactly what needs to be changed if new types of materials are added to the model or new functionalities that require that creation are added.

Your task then for this seminar is to create a Simple Material Factory class to encapsulate that creation. There is no need at this point to implement this as a method of an application class (Factory method - that will come later! CW2 ☺). Just make sure that you call the Simple Material Factory creation method to create your objects when you load them from a file or when you are adding new ones to the collection.