PiD: Exercise 1

Note: this exercise has been used previously by a company in Sweden to assess Java programming skills!

Remarks

Implement as much of the specified functionality as you can. We will assess all submitted solutions, regardless of their level of completion. Submitting a partial solution is better than not submitting anything!

Description

You are given a text file ("animals.csv") where each line contains a list of values that are separated by commas (comma-separated values or CSV). The text file contains the data of animals. Each line contains the type (or breed), the name, and the year of birth. Here are the first three lines of the text file:

```
golden retriever,lucy,2011
dolphin,winter,2003
duck,donald,2010
```

Task

Write Java code with the following functionality:

- Read the CSV file line by line, and extract the type, name, and birth year from each line.
- Create a class "Animal" as well as several subclasses for the different types of animals represented in the text file. The subclasses must be able to store the information contained in each line of input.
- Create an interface "Swimmer" which should be implemented by all classes of animals that can swim.
- Create a collection "animals", using generics, which stores an instance of the right subclass of "Animal" for each line of the text file.
- A line of the text file is considered invalid if the year field contains a non-numeric character (a character which is not 0-9). For each invalid line read from the file a warning message should be reported. No animals should be added to the "animals" collection for invalid lines.
- Print the contents of the "animals" collection to the console, sorted alphabetically by name.
- Write a method that allows filtering a collection of animals using a custom filter. The method should return a collection of animals that satisfy the custom filter.
- Create another collection "swimmers" which contains only those animals from the "animals" collection whose class types implement the "Swimmer" interface.

- Create a new CSV file "swimmers.csv" with the contents of the "swimmers" collection, sorted chronologically by birth date. This new CSV file should have the same format as the original file "animals.csv".
- Write unit tests using JUnit which test the functionalities needed above. Structure your code in a way that enables unit tests.
- Send the "swimmers" collection to a server (running on the same computer), which should output the contents of the received collection to the console.

Take into account the following aspects:

- Quality of the design and structure of your code.
- Proper handling of exceptions.
- Proper management of resources, such as files.
- Java 8 (or above) features should be used when suitable.
- The code should be commented.

Contents of the file "animals.csv":

```
golden retriever,lucy,2011
dolphin,winter,2003
duck,donald,2010
bengal cat,billy,20I1
chicken,johnny,2015
arabian horse,blueskin,2009
german shepherd,apollo,2O13
great white shark,jaws,2005
parakeet,cleo,2010
```