Task
solution in Java code

Mobility4You is a brand-new start-up that wants to start selling cars over the internet. Just like Amazon did for the book industry or Zalando did for the clothing industry, it wants to offer a wide variety of brands and make it easy for customers to quickly choose their favourite car. Mobility4You has also decided not to offer any diesel- powered cars, as these cars are less environmentally friendly. For now, their focus is solely on cars, but in the future, they might want to extend to electric bikes, etc. (do take that into account when designing your application!).

Specifically, Mobility4You gave you the following file format and wants you to develop an application around it.

An example of such a file looks as follows:
(obviously, you are not allowed to change the file format in any way!)

```
ELECTRIC_CAR Tesla, Model 3, 150KW, 50000Ah, 30000 euro
GAS_CAR Honda, Civic, 1.5L, 80KW, 18000 euro
HYBRID_CAR Toyota, Prius, 1.5L, 50KW, 12000Ah, 24000 euro
```

The complete file **mobility.txt** is in so-called CSV (Comma Separated Value) format and is available on the S drive (the exception to this CSV format being that there is no comma after the first element (e.g., ELECTRIC_CAR)).

The order of the properties (model, engine power, battery capacity, …) is fixed. The properties per element Electric Car, Gas-powered Car, and Hybrid Car are listed below:

An **ELECTRIC_CAR** is characterised by:
- The brand
- The model name
- The power of the engine
- The capacity of the battery
- The price

A **GAS_CAR** is characterised by:
- The brand
- The model name
- The engine displacement
- The power of the engine
- The price

A **HYBRID_CAR** is characterised by[1]:
- The brand
- The model name
- The engine displacement
- The power of the engine
- The capacity of the battery
- The price

Mobility4You asks you to design and implement a program that:

- **Reads in** the file mobility.txt
- **Stores** the read-in data in a suitable data structure
- **Outputs** the entire catalogue to the screen
- Allows to **add** new configurations of existing products (e.g., cars of different brands, cars with more powerful engines, etc.)
- Allows to **write to file** all product information (preserving the file format!).
- Write an **equals()** method for each class (except for the class that contains the main() method)
- To enable user interaction, please provide a **command line interface** with System.out.*. This interface should look like:

```
Please make your choice:
   1 - Show the entire Mobility4You catalogue
   2 – Add a new electric car
   3 – Add a new gas-powered car
   4 – Add a new hybrid car
   5 - Show the entire Mobility4You catalogue sorted by car-type
   6 – Show the entire Mobility4You catalogue sorted by brand
       (alphabetically)
   7 – Write to file
   8 – Stop the program
```

[1] A hybrid car has a traditional combustion engine and an electrical engine.

Option 1

All products are shown on screen in the same format as in the file:

```
ELECTRIC_CAR Tesla, Model 3, 150KW, 50000Ah, 30000 euro
GAS_CAR Honda, Civic, 1.5L, 80KW, 18000 euro
HYBRID_CAR Toyota, Prius, 1.5L, 50KW, 12000Ah, 24000 euro
```

Option 2, 3 & 4

- Through questions you ask the user to fill in all the necessary fields that compose an electric-powered (Option 2), gas-powered (Option 3) or hybrid (Option 4) car. Asking the user for a single line with all information is not sufficient.

Option 5&6

- Use the same file format as in Option 1 to show the catalogue sorted according to model (for example, first all electrical vehicles, then all hybrids, then all gas-powered cars – Option 5) or brand (Option 6).
- Please note that both sorting options **need to be implemented with a Comparator** (see later on in this assignment for more details)

Option 7

The data should be written to file, in the same format so that the application can read in the file again! The old file should be overwritten!

Option 8

The application stops.

Some important things to consider for this assignment:

- Think about the usefulness of applying **inheritance**.
- The **filename mobility.txt should not be hardcoded** in your Java program. Please make sure to let the user provide it when starting the program (either as an explicit question to the user or as a "command line input")
- The program should **compile**
- For a good grade, your program should also work well, without exceptions. Take care to have a nice **programming style.** In other words, make use of code indentation, whitespaces, logical identifier names, etc.

Sorting for options 5 and 6 in more detail

Mobility4You is asking you to implement two ways to sort their catalogue, namely by product (first all electrical vehicles, then all hybrids, then all gas-powered cars) and by brand (alphabetically, so first all cars from Audi, then BMW, …). It might very well be that later on, Mobility4You wants you to implement other types of sorting as well, that is why a flexible way of sorting is important.