**Exercises on File Handling**

1. We want to realize a program for handling new and used cars to sell.

For each car to sell, the information about the car is stored on a text file.

Realize a class Car, to handle a single car.

Each car is characterized by the following information:

model,

manufacturing year,

driven kilometers,

and price.

New cars can be distinguished by the fact that the driven kilometers are 0.

The class Car should export the following methods:

• a constructor to construct a car object, given all the data about the car as parameters;

• suitable get-methods to obtain the data about the car;

• a toString method, which does overriding of the toString method inherited from Object, and returns a string containing the data about the car;

• boolean equalTo(Car c) : that returns true if the car coincides with the car c in all of its data, and false otherwise.

• public static Car read(BufferedReader br) throws IOException : that returns a Car object reading from the file.

Realize a class CarList, each of whose objects represents a list of cars to sell.

A CarList object does not directly store the data about the cars to sell, but maintains the name of a text file in which such data are stored, according to the following format:

model

manufacturing year

driven kilometers

price

The class CarList should export the following methods:

• CarList(String filename) : constructor with a parameter of type String, representing the name of the file in which the data about the cars to sell are stored;

• int countNewCars() : that returns the number of new cars in the list of cars to sell;

• Car mostExpensiveCar() : that returns the Car object corresponding to the most expensive car in the list of cars to sell.

• void addCar(Car c) : that adds the car c to the end of the list of cars to sell.

• void remove(Car c) : that removes from the list of cars to sell the car whose data coincides with that of c, if such a car is present, and leaves the list unchanged otherwise.

1. A **PhoneDirectory** holds a list of names and associated phone numbers. But a phone directory is pretty useless unless the data in the directory can be saved permanently -- that is, in a file.

Write a phone directory program that keeps its list of names and phone numbers in a file.

The user of the program should be able to look up a name in the directory to find the associated phone number.

The user should also be able to make changes to the data in the directory.

Every time the program starts up, it should read the data from the file.

Before the program terminates, if the data has been changed while the program was running, the file should be re-written with the new data.