Mark 2:

Create the interface Vehicle with the method getDetails() returning String, must be inserted

Mark 3:

a. Create public class Car which must implement Vehicle interface and It must be inserted 3 private fields: **weight - float, price - double, model – String**

Mark 4:

b. for this class fields, it is mandatory to implement getters and setters , plus the default constructor (without parameters), **there is NO constructor with parameters**

c. override implementation for the getDetails() method (from the Vehicle interface) to return the model String

d. the setters should throw Exception if the constraints are not fulfilled:

- model different than null and model String length greater than 1
- price and weight greater than 0 each
- otherwise throws Exception

e. Implements Serializable, Cloneable and Override the implementation for equals(), hashCode() and clone() methods in the right way.

Mark 5:

a. create public class ElectricCar must inherit the Car class and it adds the batteryLife - int field

b. for this class fields it is mandatory to implement getters and setters

c. override virtual implementation for the getDetails() method (from the Vehicle interface and Car implementation), to return the batteryLife as a String

d. the setters should throw Exception if the constraints are not fulfilled:

- batteryLife greater than 0

Mark 6:

Create public class Utils which contains private static list field with interface as type: List<Vehicle>

Insert the following methods:

a. public static List<Vehicle> createCars(int n) throws Exception - for creating an ArrayList of n elements which are containing n default Car objects and it using the static field of the class (list)

b. public static List<Vehicle> readCars(String file) - for reading and parsing text files with string lines for creating Car objects (e.g. please see for example carsList.txt file); first line is the weight in kg, second line is price in E and third line is the model)

hint: use RandomAccessFile and read / parse line by line (first is parsing for float - weight, second line is dou rice, third is String - producer)

Mark 7:

a. public static void writeBinaryCars(String file, List<Vehicle> listP) - for writing binary the cars into the fi hint: use FileOutputStream with FileOurputStream to serialized/save the Car objects from the ArrayList of the ca bjects

b. public static List<Vehicle> readBinaryCars(String file) - for reading binary the Car objects from the file eating the ArrayList

Mark 8:

a. create the class VectThread which implements Runnable and contains 2 fields:

- carsList with interface as type List<Vehicle>
- avgWeight is a real (double) number for storing the average weight of the cars list

b. In the constructor read the file using readBinaryCars static method from Utils

c. provide get methods for the both fields (carsList and avgWeight) of the class
d. Within the override run method (with the signature in Runnable interface)
- the developer should go through the carsList and calculate the average of the weights from the car list objects (Car class - explicit cast)

Mark 10:

Create public class UtilsDAO with only one static field c from class Connection (SQL/JDBC)

This class contains the following 3 static methods:

a. public static void setConnection() - set the connection by using org.sqlite.JDBC driver and connection string: jdbc:sqlite:test.db

b. public static void closeConnection() - close the SQL/JDBC connection

c. public static String selectData() throws SQLException - for SQL selecting all the cars from the already created SQLite DB file

- select * from CARS
- CARS table is already created with the following columns id - INT, MODEL - TEXT, PRICEAL and WEIGHT - REAL
- the String containing the view after selecting the entire table is having each table line separith "\r\n" and each column value will be separated by ":"