**SUBIECTE 2018**

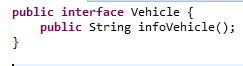
Digital specifications (specs) for the examination - **Subject 12:**

**//Mark 3:**

**//**Create the interface **Vehicle** with the method infoVehicle() returning string, must be inserted

***>*** *File -> New Java Project*

***>*** *Click dreapta pe proiect -> New -> Interface*



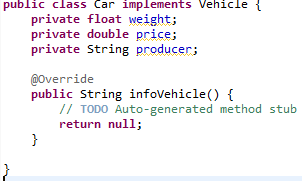
**//Mark 3:**

**//a.** Create public class **Car** which must implement Vehicle interface and it must be inserted 3 fields:

***>*** *Click dreapta pe proiect -> Add -> Class*

***>*** *La interfaces cauti si selectezi interfata Vehicle*

**//**weight - float, price - double, producer - String



**//b.** for this class fields, it is mandatory to implement getters and setters, plus the default constructor (without parameters),

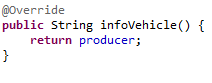
**//**there is NO constructor with parameters

**>** *Source -> Generate Getters and Setters*

***>*** *Plus default constructor*

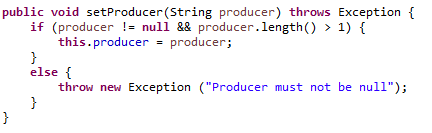
**

**//c.** override virtual implementation for the infoVehicle() method (from the Vehicle interface) to return the producer String

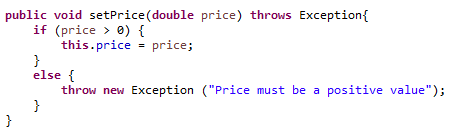


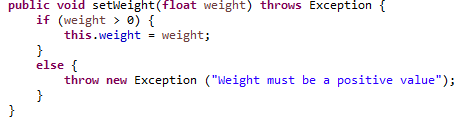
**//d.** the setters should throw Exception if the constraints are not fulfilled

**//**-producer different than null and producer 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

**>** *scrii asta*

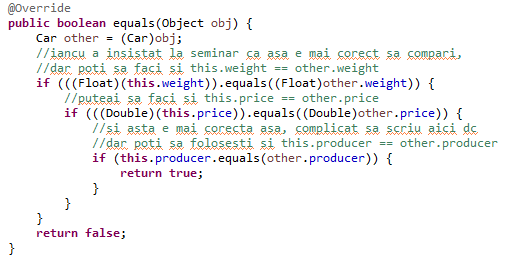
**

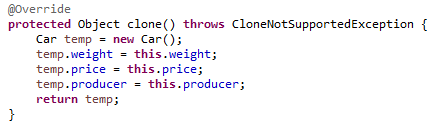
***>*** *importi ce trebuie la Serializable*

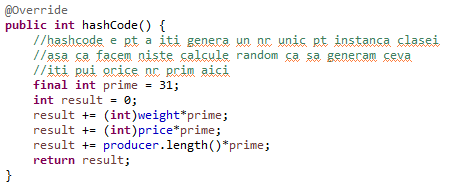
***>*** *tii mouse-ul deasupra la Car, apesi pe generate serial version id*

***>*** *teoretic, aici puteai sa apesi pe Source - > Generate Equals and HashCode si sa folosesti functiile alea blanao, numai ca nu stim daca Toma o sa fie de acord sau nu. Asa ca facem manual functiile astea meanwhile*

***>*** *Ctrl + Space si cauti equals, hashcode si clone ca sa implementezi functiile*

**

**

**

**//Mark 3:**

**//a.** Create public class AutonomousCar must inherit the Car class and it adds the softwareVersion - int field

***>*** *Click dreapta pe proiect -> Add new class -> La Superclass cauti si selectezi Car*

**

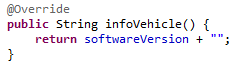
**//b.** for this class fields it is mandatory to implement getters and setters

***>*** *Source -> Generate Getters and Setters*

**//c.** override virtual implementation for the infoVehicle() method (from the Vehicle interface and Car implementation),to return the softwareVersion to a String

***>*** *tii mouse-ul deasupra la AutonomousCar, apesi pe generate serial version id*

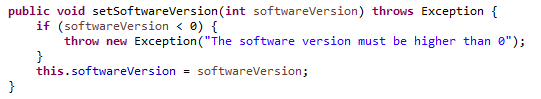
***>*** *Ctrl + Space, cauti infoVehicle*

**

***>*** *daca adunam un int la un string, return-ul va fi tot un string. e ca un cast, dar mai blanao*

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

**//** - softwareVersion greater than 0

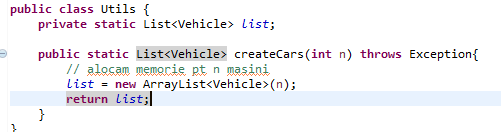


**//Mark 4:**

// Create public class Utils which contains private static list field with interface as a 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)



***>>>*** *scroll down for more*

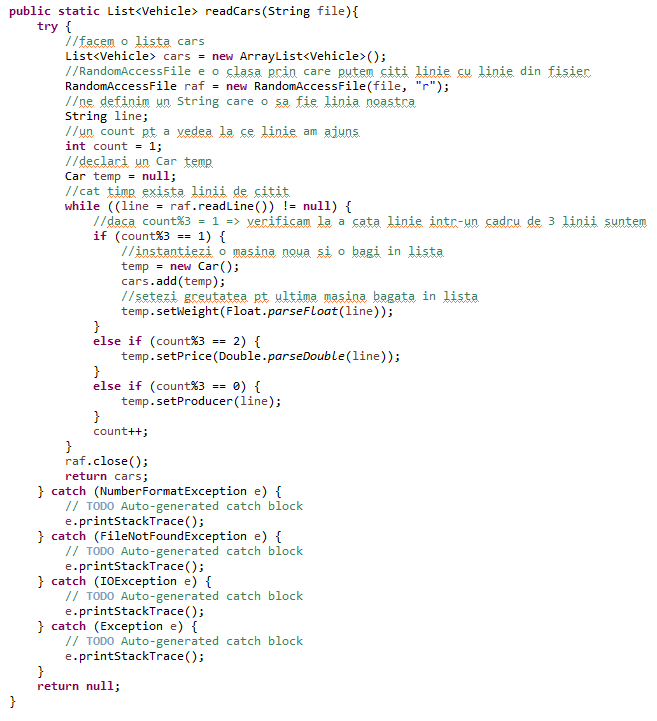
// **b.**  public static List<Vehicle> readCars (String file)

// - for reading and parsing text files with string lines for creating Car objects

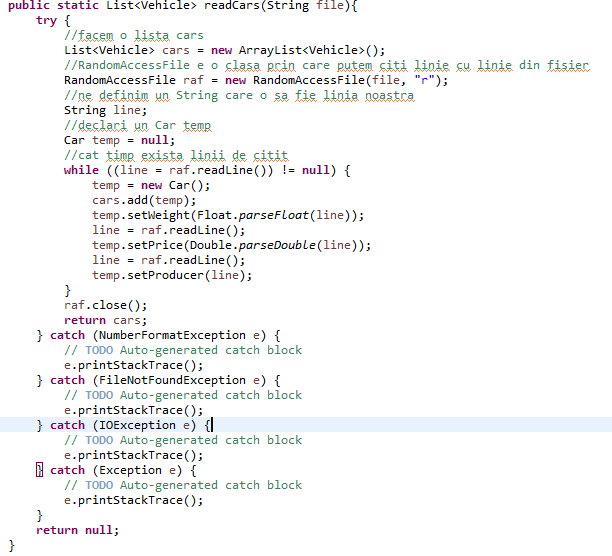
// (e.g. please see for example carsList.txt file); first line is the weight in kg, second line is price in EUR and third line is the producer

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

***>*** *cand scrii codul aici ar fi mai simplu sa scrii tot fara try catch, dupa sa selectezi tot ce ai scris in functie, click dreapta -> surround with try catch block*

**

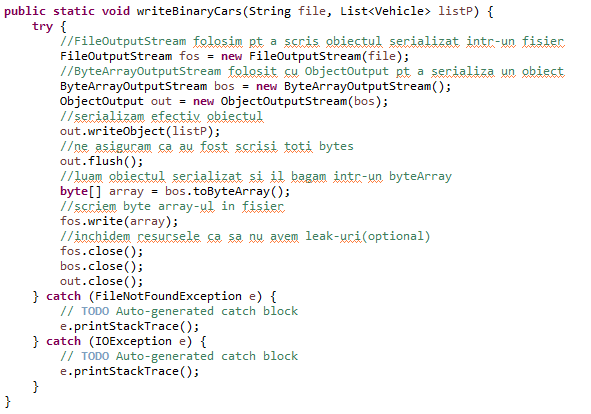
**SAU**

****

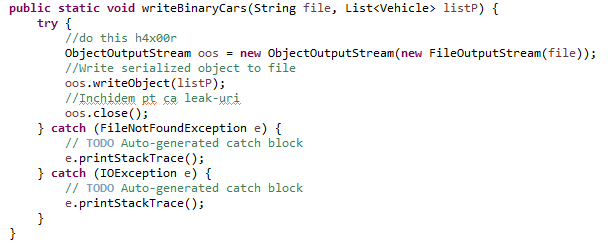
***>>>*** *scroll down for more*

// **c.** public static void writeBinaryCars(String file, List<Vehicle> listP) - for writing binary the cars into the file

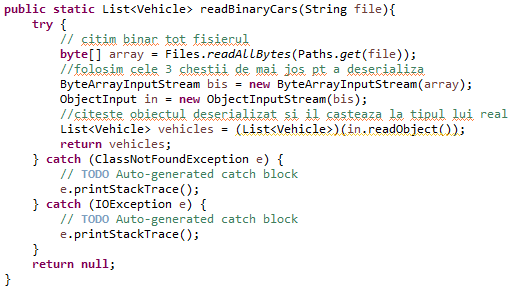
// hint: use FileOutputStream with FileOutputStream to serialized/save the Car objects from the ArrayList of the cars objects



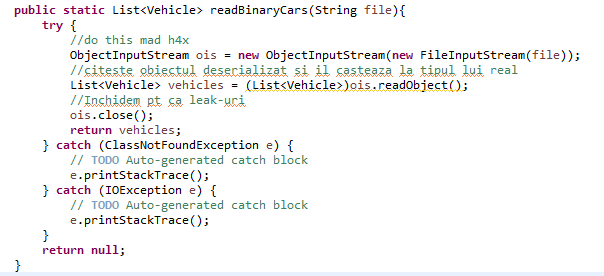
**SAU**

****

//**d.** public static List<Vehicle> readBinaryCars(String file) - for reading binary the Car objects from the File and creating the ArrayList



**SAU**

****

**//Mark 4**

// Beside the JUnit 4 framework for running for the official mark,

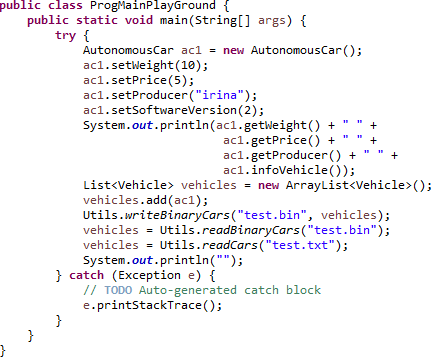
// please do your minimal testing - e.g. create an instance from AutonomousCar and call methods in main method from ProgMainPlayGround class

// createCars or readCars from Utils class, sockets objects, etc.

// DO NOT spend too much time here - No requirements and a lot of testing code is already in JUnit4 Framework

***>*** *click dreapta pe proiect, add new txt file cu numele test.txt*

***>*** *iti scrii in txt valorile*

**

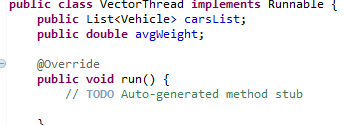
**//Mark 5:**

**//a.** Create the class VectorThread which implements Runnable and contains 2 fields:

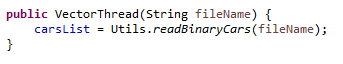
**//** -CarList with interface as type List<Vehicle>

**//** -avgWeight is a real(double) number for storing the average weight of the cars list

***>*** *Click dreapta pe proiect -> Add new class -> Pui numele VectorThread -> La interfaces cauti si selectezi Runnable*

**

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

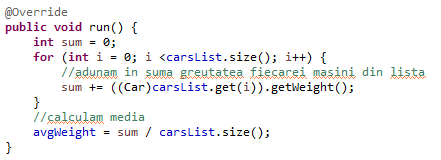


**//c.** provide get methods for the both fields (argList and avgWeignt) of the class

***>*** *Source -> Generate Getters and Setters -> Selectezi doar getteri*

**//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)



**//Subject of +2 points ⇔ Mark 6 or 7(and parts of mark 8):**

**//a.** Create public class TCPServerSocketMultiT which handles multi-threading TCP server socket connections for implementing a proprietary communication protocol (set of rules)

**//b.**The class contains the following private fields:

-serversocket as ServerSocket,port =50001 as int,f as File and vt as VectorThread(“has a” relationship)

**//c.**The class contains the folllowing methods and constructor:

**//c1.** constructor which get the port as parameter and create the serverSocket:

public TCPServerSocketMultiT(int port) throws Exception

**//c2.** getter and setter for the field port

**//c3.** public void SetFileName(String fileName) method which allocate memory for the field f, if and only if, the string parameter is different than null, otherwise is setting null

**//c4.** public void startTCPServer() throws IOException method which is having the infinite processing loop and is immplementing 3 commands from the propietary pro…**???**

**Hints** for startTCPServer method:

**-** create multi-threading by using lambda expressions from Runnable functional interface after the blocking accept() method from serverSocket object.

**-** get the input stream as BufferedReader and output stream as ObjectOutputStream

**-** initialize the vt field from class VectThread by passing the file absolute path from f field as parameters AND OBTAIN the list (ArrayList of Car objects) from the … **???**

**-** parse line by line the TCP request

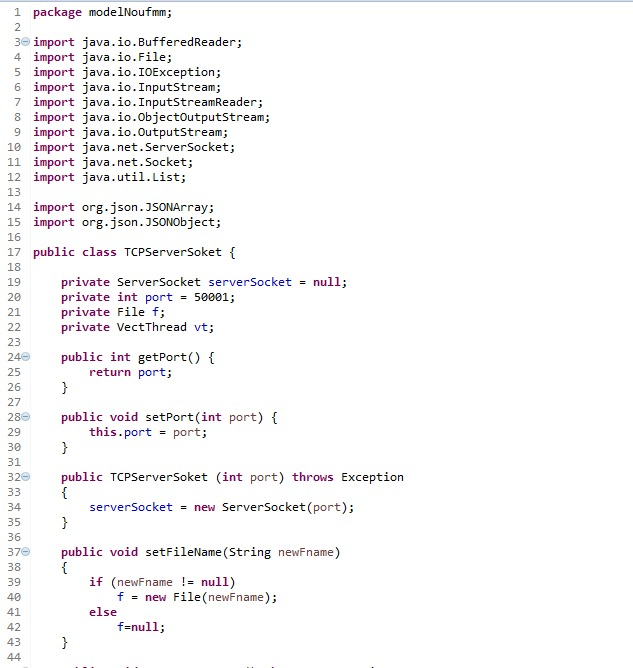
**-** if EXIT text command is recived over the socket,then break the infinite loop of the processing and send TCP FIN packet back to the TCP client (e.g. by **???**)

**- (Mark 6)** if GETFILE text command is recived over the socket,then reply back the serialized list encapsulated in the vt object field.

**- (Mark 7)** if GETJSON text command is recived over the socket, then reply back with the list in JSON format.

**- (Mark 8)** if GETDB text command is recived over the socket, then reply back with the list as String produced by  **???** UtilsDAO.selectdata() /UtilsDAD.selectdata() (please also take into  **???**)

**//**YOU MAY NOT create the TCP client because it is already created into JUnit test framework; for mark 8, please also see UtilsDAD / UtilsDAO **???** class (without UtilsDAD/UtilsDAO **???** class, mark 8...)







Pentru 7

**Subject 11**

**Mark 3**: Create the interface ElectronicDevices with the method InfoDevice(), which is returning String must be inserted.

***>*** *File -> New Java Project*

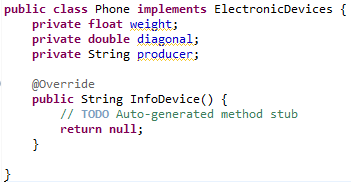
***>*** *Click dreapta pe proiect -> New -> Interface*

**

**Mark 3**:  
**a**. Create public class Phone which must implement ElectronicDevice interface and it must be inserted 3 fields: weight-float; diagonal-double; producer-String

***>*** *Click dreapta pe proiect -> Add -> Class*

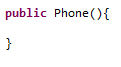
***>*** *La interfaces cauti si selectezi interfata Electronic Devices*

**

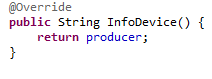
**b**. For this class fields it is mandatory to implement getters and setters, plus the default constructor (without parameters). There is NO constructor with parameters.

**>** *Source -> Generate Getters and Setters*

***>*** *Plus default constructor*

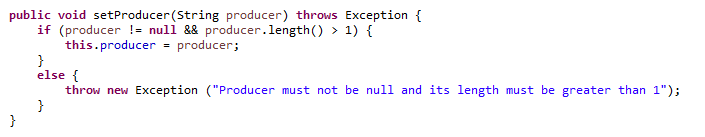
**

**c**. Override implementation for the InfoDevice() method (from the ElectronicDevices interface) to return the producer String

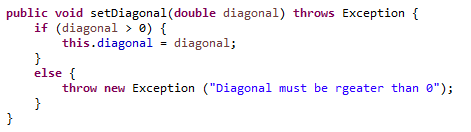


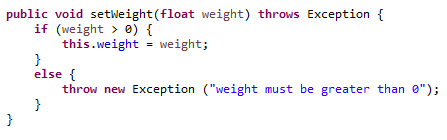
**d**. The setters should throw exception if the constraints are not fulfilled:

-producer different than null and producer string length greater than 1;



-diagonal and weight greater than 0 each.





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

**>** *scrii asta*

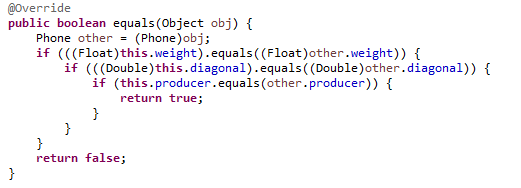
**

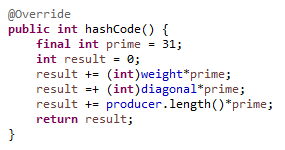
***>*** *importi ce trebuie la Serializable*

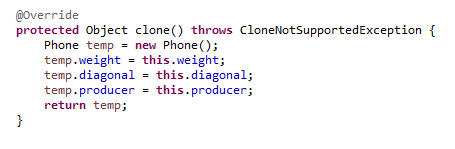
***>*** *tii mouse-ul deasupra la Phone, apesi pe generate serial version id*

***>*** *teoretic, aici puteai sa apesi pe Source - > Generate Equals and HashCode si sa folosesti functiile alea blanao, numai ca nu stim daca Toma o sa fie de acord sau nu. Asa ca facem manual functiile astea meanwhile*

***>*** *Ctrl + Space si cauti equals, hashcode si clone ca sa implementezi functiile*

**

**

**

**Mark 3**:  
**a**. Create public class SmartPhone which inherits the Phone class and it adds the batteryDuration - int field

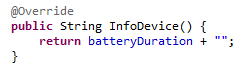
***>*** *Click dreapta pe proiect -> Add new class -> La Superclass cauti si selectezi Phone*

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

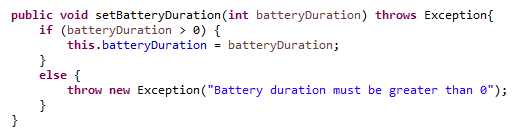
***>*** *Source -> Generate Getters and Setters*  
**c**. Override virtual implementation for the InfoDevice() method (from the ElectronicDevices interface and Phone implementation), to return the batteryDuration as a String

***>*** *tii mouse-ul deasupra la SmartPhone, apesi pe generated serial version id*

***>*** *Ctrl + Space, cauti InfoDevice()*

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

-batteryDuration greater than 0.

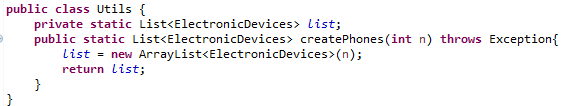


**Mark 4:**

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

- Insert the following methods:

1. public static List<ElectronicDevices> createPhones[int n] throws Exception - for creating an ArrayList of n elements which contains n default Phone objects and uses the static field of of the class (list)



1. public static List<ElectronicDevices> readPhones(String file)

* for reading and parsing text files with string lines for creating Phone objects

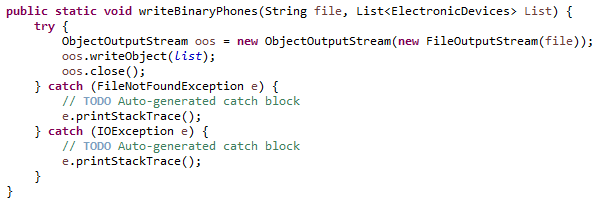
(e.g. Please see for example phonesList.txt file); first line is the weight in grams, second line is screen diagonal and 3rd line is the producer

***>>>*** *scroll down for more*

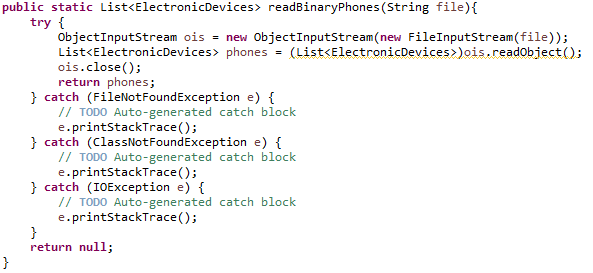


c. public static void writeBinaryPhones(String file, List<ElectronicDevices> List) for writing binary the phones into the file

hint: use FileOutputStream with **FileOutputStream (! aici a zis Petrut ca probabil ar fi trebuit sa scrie ObjectOutputStream )** to serialize/ save the Phone objects from the ArrayList of the phones objects



d. public static List<ElectronicDevices> readBinaryPhones(String file) - for reading binary the Phone objects from the file and creating the ArrayList



**Mark 4:**

Beside the JUnit 4 framework running for the official mark, please do your own minimal testing: e.g: create an instance from SmartPhone and call methods in main method from ProgMainPlayGround class

-createPhones or readPhones from Utils class, sockets objects, etc.

-DO NOT spend to much time here! Requirements and a list of testing code is already into JUnit4 framework

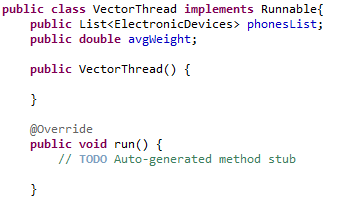


**Mark 5:**

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

-phonesList with interface of type List<ElectronicDevices>

-avgWeight is a real (double) number for storing the average weight of the phones list



b. in the constructor read the file using readBinaryPhones static method from Utils



c. provide get methods for the both fields (phoneList and avgWeight) of the class

***>*** *Source -> Generate Getters and Setters -> Selectezi doar getteri*

d. within the override run method (with the signature in Runnable interface) the developer should go through the phoneList and calculate the average of the weights from the phone list objects (Phone class - explicit cast)

