LAB 211 Assignment

Type:	**Long Assignment**
Code:	**J1.L.P0027**
LOC:	**400+**
Slot(s):	**N/A**

Title

The Vehicle Management - Read and Write File.

Background

- **The tranport company** wants to develop an application to **manage the vehicles at their showroom**. Vehicles have **many features and properties**. Every vehicle has **some attributes** such as **ID_Vehicle, Name_Vehicle, color_Vehicle, price_Vehicle, brand_Vehicle, type, productYear.**

- This application needs to have **some core functions** such as: **display** Vehicles information, **search** Vehicle, **add** Vehicle, **edit** Vehicle…

- Vehicle information is stored in a text or binary file (**vehicle.dat**).

- The application **must be built** based on **OOP model.**

Program Specifications

Build a vehicle management application with the **following basic functions**:

1. **Add** new vehicle.
2. **Check** exits Vehicle.
3. **Update** vehicle.
4. **Delete** vehicle.
5. **Search** vehicle.
   5.1 Search by ID_Vehicle.
   5.2 Search by Name_Vehicle.
6. **Display** all Vehecle
7. **Save** all vehicles to file.
8. **Print all** vehicle **from** the **file**.

Others: **Quit**

Features:

This system contains the following functions:

The application **displays a menu** and **wait for user to select** an option.

- **Function 1: Add new vehicle – 50 LOC**
  - The application requires the user input vehicle information. Some information is included such as **ID_Vehicle, Name_Vehicle, color_Vehicle, price_Vehicle, brand_Vehicle, type, productYear.**
    - **Notes**: The **valid data of fields** must be **checked**.
  - The application will **add the new vehicle** and **show the result** of the add **with the success or fail message.**
  - **The application asks** to **continuous** create new vehicle **or go back** to the main menu.

- **Function 2: Check to exist Vehicle – 50 LOC**
  - The application will **check the ID_Vehicle** that is **stored** in the **vehicle.dat file**.
  - **A "Existed Vehicle" message** should be **displayed** whether the **ID_Vehicle has existed** in the vehicle.dat file.
  - **Otherwise**, the **"No Vehicle Found!" message** will display.
  - The application should **go back** to the main menu and **wait for asking user to choose one option**.

- **Function 2: Update vehicle – 50 LOC**
  - The application **requires** the user input the **vehicle's id_Vehicle**.
  - If vehicle **does not exist**, the notification **"Vehicle does not exist" is shown**. **Otherwise**, user can **input update information** of vehicle to update vehicle.
  - If new **typed information** is **blank**, then **no information is changed**.
    - ✓ **Notes** new information must be validated (similar to add new vehicle information).
  - System **must print out** the result of the updating.
  - **After updated**, the application **returns to the main screen and wait for asking user to choose any option**.

- **Function 3: Delete vehicle – 50 LOC**
  - User can **delete any vehicle** in the showroom **by id_vehicle**.
  - Before the **delete** action is **executed**, the **system** must **show confirm message**.
  - The **result** of the delete action **must be shown** with **success or fail message**.
  - **After deleted**, the application **returns to the main screen and wait for asking user to choose any option**.
- **Function 4: Search vehicle**
  - Student creates a submenu that allows the user to select 02 ways: **search vehicle by Name or search vehicle by Id.**

✓ **S.4.1: Search by Name – 50 LOC**
  - User inputs the **searching string**.
  - The system will search in the show room, and it returns **all vehicle with their name** containing the search string.
  - The application will show result list with **all information** of vehicles that are **sorted by name descending**
  - The application **returns to the main screen and wait for asking user to choose any option**.

✓ **S.4.2: Search by Id – 50 LOC**
  - The user **enters any Id**.
  - The system searches in the show room, and it returns the vehicle that has **id matching the**

search string.
- The application will show the result with **all information** of vehicle
- The application **returns to the main screen and wait for asking user to choose any option**.

- **Function 5: Display vehicle list**
  - Student creates a submenu that allows the user to select 02 ways: show all or show by price.

✓ **D.5.1: Show all – 50 LOC**
- The system will show all vehicles with their information in the show room.
- **After shown**, the application **returns to the main screen and wait for asking user to choose any option**.

✓ **D.5.2: Show by price – 50 LOC**
- The user enters any price.
- The system searches in the show room, and it returns all vehicles' information that their price **less than inputed price**. The result list will be **sorted by price_vehicle field descending**
- **After shown**, the application **returns to the main screen and wait for asking user to choose any option**.

- **Function 6: Save data to file – 50 LOC**
  - The application will **write the vehicles' information list** to the file (**vehicle.dat**).
  - The application **returns to the main screen and wait for asking user to choose any option**.
- **Function 7: Print vehicle list**
  - Student creates a submenu that allows the user to select 02 ways: **print all** or **print by year**.

✓ **P.7.1: Print all – 50 LOC**
- **The system will display all vehicles with their information of the show room.**
- **After shown**, the application **returns to the main screen and wait for asking user to choose any option**.

✓ **P.7.2: Print by Year – 50 LOC**
- The user enters any year.
- The system searches in the show room, and it print all vehicles' information that their year **greater equal than inputed year**. The result list will be **sorted by product Year field descending**
- **After shown**, the application **returns to the main screen and wait for asking user to choose any option**.

- **Bonus 50 LOC (not over maximum 500 LOC in total of this project) if the student applies one of the Design Patterns (such as DAO pattern, Factory pattern, Repository pattern, and so on) in this project. More references for the design pattern:** https://www.tutorialspoint.com/design_pattern/index.htm
- The above specifications are **only basic information**. You must perform a **requirements analysis step and build the application according to real requirements.**
- The **lecturer** will **explain the requirement only once on the first slot of the assignm**