LAB 211 Assignment

Type: Long Assignment
Code: J1.L.P0025
LOC: 500
Slot(s): N/A

Title

Store Management at Convenience Store

Background

A convenience store sells 2 groups of products: products for daily use (bread, dumplings, sticky rice, ...) and products with long shelf life such as cookies, soft drinks, filtered water, ... The **product.dat** file is used to store information of all products in the Store. The **wareHouse.dat** file has stored import/export information. **Each import/export receipt can contain many different items with different quantities.**

Program Specifications

Build a management program with the following basic functions

1. **Manage products**
    - 1.1. Add a product
    - 1.2. Update product information.
    - 1.3. Delete product.
    - 1.4. Show all product.
2. **Manage Warehouse**
    - 2.1. Create an import receipt.
    - 2.2. Create an export receipt.
3. **Report**
    - 3.1. Products that have expired
    - 3.2. The products that the store is selling.
    - 3.3. Products that are running out of stock (sorted in ascending order).
    - 3.4. Import/export receipt of a product.
4. **Store data to files**
5. **Close the application**

Each menu choice should invoke **an appropriate function** to perform the selected menu item. Your program must **display the menu**. **After each task**, the application should **wait** for the **user to select another option** until the user chooses to quit the program.

Features:

This system contains the following functions:

Display a menu and ask users to select an option.

1. **Build your data structure (80 LOCs)**

- **Product information** includes some fields such as product **code**, product **name**, **manufacturing date**, **expiration date**, and **other** attributes.
- **Warehouse information** includes some **required fields** such as the **import/export code** *(a self-incrementing 7-digit number*), the **time** to create the import/export slip (*it should be taken from the system time*). **Each code** can **contain many different items** with different quantities.
- Classes, abstract classes, Interfaces **are required in building** this application.
- **Student must implement** the **polymorphism properties** of object-oriented programming.

2. **Manage products/items of the store:**

1.1. Add a product. (30 LOCs)

- ✓ **Create** a submenu.
- ✓ **Add** the new product **to collection**.
  - **Notes**: **product code can not duplicate, check valid information of quantity and manufactoring/expired date, …**
- ✓ **The application asks** to **continuous** create new product **or go back** to the main menu.

1.2. Update product information. (50 LOCs)

- ✓ **User requires** enter the **productCode**
- ✓ If product code **does not exist**, the notification "**Product does not exist**" message is shown. **Otherwise**, user can **input update information of product** to update that product.
- ✓ If new **typed information** is **blank**, then **no information is changed**.
  - **Notes** new information must be validated (similar to add new product 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**.

When changing product information, the corresponding information must be adjusted on the import/export receipts.

1.3. Delete product. (20 LOCs)

- ✓ 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**.

Note: the application should only remove the product from the store's list when the import / export information for this product has not been generated.

1.4. Show all product. (20 LOCs).

- ✓ The application **shows all** data in the **product.dat file** or **product's collection** into the screen.
- ✓ The application **returns to the main screen and wait for asking user to choose any option**

3. Warehouse management

2.1. Create an import receipt. (50 LOCs)

- ✓ **Create** a submenu.
  - **Notes**: **import code can not duplicate, check valid information of quantity and manufactoring/expired date, …**
- ✓ **Add** the new product **to** the **import receipt**.
- ✓ **Ask to continuous** add new product **or show confirm message**.
- ✓ **After confirmed**, the result of the import receipt **must be shown** with **success or fail message**. Then, the application **returns to the main screen and wait for asking user to choose any option**.

2.2. Create an export receipt. (100 LOCs)

- ✓ **Create** a submenu.
  - Notes: **export code can not duplicate**, **check valid information of quantity and manufactoring/expired date, …**
- ✓ **Add** the new product **to the export receipt**.
- ✓ **Ask to continuous** add new product **or show confirm message**.
- ✓ **After confirmed**, the result of the export receipt **must be shown** with **success or fail message**. Then, the application **returns to the main screen and wait for asking user to choose any option**.

4. Report

4.1. Products that have expired. (25 LOCs)

- ✓ The application **searches** the **product list**, and it **returns all product** that has manufacturing date **greater than** expiration date.
- ✓ The result list is shown with some information as **product code, product name, production date, expiration date, quantity**

- ✓ The application **returns to the main screen and wait for asking user to choose any option**.

4.2. The products that the store is selling. (25 LOCs)

- ✓ The application **searches** the **product list**, and it **returns all product** that has quantity **larger than** 0 and manufacturing date **less equal than** expiration date.
- ✓ The result list is shown with some information as **product code, product name, production date, expiration date, quantity**
- ✓ The application **returns to the main screen and wait for asking user to choose any option**.

4.3. Products that are running out of stock (25 LOCs)

- ✓ The application **searches** the **product list**, and it **returns all product** that has **quantity less equal than 3**.
- ✓ The result list is shown with some information as **product code, product name, production date, expiration date, quantity**
- ✓ The result list should be sorted in **ascending order by quantity**
- ✓ The application **returns to the main screen and wait for asking user to choose any option**.

4.4. Import/export receipt of a product. (25 LOCs)

- ✓ User must **enter the product code**
- ✓ If product code **does not exist**, the notification "**Product does not exist**" message is shown.
- ✓ **Otherwise**, the application **shows product's data** in **the warehouse.dat file** or **warehouse's collection** into the screen.
- ✓ The application **returns to the main screen and wait for asking user to choose any option**.

5. Store data to files. (50 LOCs)

- ✓ **The list product** should be **stored** into the **product.dat file**.
- ✓ The **list warehouse information** should be **stored** into the **warehouse.dat file**.
- ✓ The application **returns to the main screen and wait for asking user to choose any option**

- 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 assignment.