LAB 211 Assignment

Type:	**Long Assignment**
Code:	**J1.L.P0019**
LOC:	**500**
Slot(s):	**N/A**

Title

Flower Store Management

Background

Write a program to **manage a flower store**. The program implements the terminology of the Object-Oriented Programming (OOP) paradigm. OOP is one of the best choosing ways to design software programs.

In this assignment, we will use the **Set<Flower> structure** to store a collection of objects since it is a unique flower species.

Program Specifications

Build the **Flower Store project** with the **menu** as follows:

Manage flower

1. **Add** a flower
2. **Find** a flower
3. **Update** a flower
4. **Delete** a flower

Manage Order

5. **Add** an order
6. **Display** orders
7. **Sort** orders
8. **Save** data
9. **Load** data
10. **Quit**

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

Features:

Function 1. Add a flower – 50 LOC

- Require to input a pet: **id, description, import date, unit price,** and **category**.
- **The system should check the valid data** with the following conditions:
  - **All** fields are **not allowed null**.
  - The **id** field must be **unique**.
  - The **length of the description** field must be from **3 to 50 characters.**
  - The import **date** field must be a **valid date format**.
  - The **unit price** field must be a **positive number**.

- **Add** the flower **to the collection of flowers**.
- **Ask** to **continue adding** a new flower or **go back** to the **main menu**.

Function 2. Find a flower – 50 LOC

- User is **required** to **input** the **flower id** or flower **name**.
- If the flower **does not exist**, the message "**The flower does not exist**" is displayed. **Otherwise**, display the flower.

Function 3. Update a flower – 50 LOC

- User is **required** to input the flower **name**.
- If the flower **does not exist**, the message "**The flower does not exist**" is displayed. **Otherwise**, the user can edit the flower.
- Show the result of the update: **success or failure**.

Function 4. Delete a flower – 50 LOC

- User is **required** to input the flower **id**.
- If the flower **does not exist**, the message "**The flower does not exist**" is displayed. **Otherwise**, the user can delete the flower.
- The **screen must show** the **confirmation message** before deleting.
- The flower **cannot be deleted** if it **has already existed** in an order detail.
- Show the result of the deletion: **success or failure**

Function 5. Add an order – 50 LOC

- An order includes an **order header and several order details**. The order header includes **the order id, the order date**, and the **customer's name**. The order **detail** includes the **order detail id, the flower id, the quantity**, and the **flower cost**.
- The system should **check the valid data** with the following conditions:
  - **All** fields are **not allowed null**.
  - The **id** fields must be **unique**.
  - The order **date** field must be a **valid date format**.
  - The **quantity** field must be a **positive integer**.
  - The flower **cost** field is the field calculated as follows**: flower cost = quantity x unit price**.
- **Add** the order to the **collection of orders**.
- **Ask** to **continue adding** a new order or **go back** to the main menu.

Function 6. Display orders – 50 LOC

- User is **required** to input a **start and end date**.
- The screen should **show orders** base on **inputted date range** as below if applicable.

LIST ORDERS FROM 10/01/2022 TO 10/31/2022

No.	Order Id	Order Date	Customer	Flower Count	Order Total
1	0006	10/01/2022	John Smith	3	$ 160
2	0007	10/15/2022	Bill Jamie	5	$ 220
3	0008	10/26/2022	John Smith	2	$ 60
	Total			10	$ 440

Function 7. Sort orders – 50 LOC

- User is **required** to input a **sorted field** (order **id** or order **date** or customer **name** or order **total**) and the sort **order** (ASC, DESC).
- Sort and display the collection of orders as below.

LIST OF ORDERS

Sorted by : Order Date
Sort order : ASC

No.	Order Id	Order Date	Customer	Flower Count	Order Total
1	0006	10/01/2022	John Smith	3	$ 160
2	0007	10/15/2022	Bill Jamie	5	$ 220
3	0008	10/26/2022	John Smith	2	$ 60
	Total			10	$ 440

Function 8. Save data – 50 LOC

- The system **saves** the **collection of flowers to** the **binary file that named with flowers.dat.**
- The system **saves** the **collection of orders** to the **binary file that named with orders.dat**.

Function 9. Load data – 50 LOC

- The system **loads** the **collection of pets** from the **flowers.dat file**.
- The system **loads** the collection of orders to **orders.dat file**.

Function 10. Quit – 50 LOC

- **Exit** the program.
- The application **must show the confirmation** message before exiting.
- The application **must save data** to files if **data has changed**.

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 in the first slot of the assignment.