ITSC 1213 – Project 2 – Pet Store Management System Extended

The main idea behind this activity is to extend the concepts we covered in Project 1. Create a system that allows a pet store owner to keep track of their inventory and members utilizing inheritance.

In this checkpoint, you will be expanding on the requirements from the previous assignment in two significant ways. This assignment gives you a working version of the Pet store project that satisfies the requirements of project 1. If you were not able to complete that assignment, or if your project had significant issues that affected functionality, you can use this one as a starting point.

Recap

The same rules apply from Project 1:

The owner of the store has asked for your help designing a system that allows them to keep track of the store's inventory and members. The owner provides you with the following details about what their store currently offers:

- The store sells three types of pets: dogs, cats, and exotic pets.
- The store offers two types of memberships to customers: regular memberships and premium memberships. The regular membership is free, while the premium members pay a fee every month.
- The store keeps track of whether the monthly fee is due or has been paid for their premium members
- The store keeps track of the members and their transactions (number of pets purchased, total purchases, ...) regardless if they are regular or premium members.
- The store also keeps track of the inventory of each pet.

The owner would like to replicate as much of this as possible but understands that software development is an iterative process and we need to start with a small set of features to implement. As as starting point you need to design a system that includes only three distinct functionality. A function, in this context, are abstract ideas like "register a new premium member", or "complete a purchase". While in many programming contexts a specific function refers to what we refer to as methods in Java, here function and method are NOT synonymous. Functions here refer to the holistic application/program/system that we are creating for managing a bookstore. To

implement your functions it may take one or more methods and will likely involve 1 or more of your classes.

One of these functions MUST be to complete a purchase. In order for that to happen the user should be able to select one or more items, and those items should be deducted from the inventory after the purchase is completed. The other two functions are entirely your choice.

As you are thinking about these features, think of any application or a website you use to purchase something online. What functionality, you as the user, are able to complete using this application? Answering this question should give you plenty of options to choose from.

You are welcome to program or map out additional functions. These do NOT necessarily need to be implemented, but can be included using empty methods (stub methods) to make your system look more complete.

Important Considerations

- We want you to take an Object-Oriented approach in this project. That means you will be creating a good number of classes that will all interact with each other, rather than having everything contained within one class and one main method. The FastFoodKitchen lab is a good example of this kind of design. In the same way that the FastFoodKitchen contained a list of Orders, the Pet store may contain a list of members, or a list of pets that can be purchased.
- It is very important to consider what attributes a book would need, that a member would need, etc. This is especially important when you consider functionalities like purchasing... which of these attributes do you need to access during a purchase, and if you need that information where/how can you find it?

Preliminaries

You can use PetStoreCode-1.zip to create a NetBeans project that has the purchase and registration working in this implementation of a Pet Store Management system that meets the requirements of project 1.

The supplied code is not a NetBeans project. Unzip the file. Create a NetBeans Project. Create the corresponding files and copy the content form the supplied code to your project. *

For Project 2 you will be including inheritance including: abstracts, interfaces, overrides, and super calls and adding functionality to the pet store management system.

Part A – Inheritance, Abstraction and Interfaces (10 points)

In this part we want to introduce inheritance and abstraction into our system.

Update this design to utilize inheritance when representing the different pets the store sells. At a minimum your design should incorporate an abstract class that is sub-classed into the three different pets available for purchasing - dogs, cats, and exotic pets.

This class must be an abstract class that implements the **Comparable** interface. When comparing pets we want that to be either based on the price of the pet or the number of pets of that type in stock. Choose one of those attributes and implement the **compareTo** method to work based on that field.

Part B – More Interfaces (10 points)

In this part you want to update your PetStore class to implement the following interface:

PetStoreSpecification

<<Interface>> PetStoreSpecification
+adoptionDrive(ArrayList<Object>) : void
+ inventoryValue() : double

Notice that this interface has only two methods: **adoptionDrive(ArrayList<Object>)** and **inventoryValue()**. The **adoptionDrive** method takes in an ArrayList of pets that are up for adoption. It should display each pet up for adoption along with its corresponding price. The **inventoryValue** method is used to get the total value of all products currently in stock. The Object class here refers to the parent class for Dog, Cat and ExoticPet. We expect one parent class for the three pet types but you might choose a different class inheritance hierarchy. Make sure you note your classes and their relationships clearly with your submission. You might decide to have a separate inventory management class that might be a better place to implement this interface. You are encouraged to discuss your design with the instructional team for feedback on your project design. Keep in mind that with computational problem solving and design there are many ways of achieving a solution. Explore and have fun with it!

Part C – Changing the Test Harness (6 points)

Update the Main class to include menu options to test the functionality added in Parts A (Comparing pets) and B (restock pets and display inventory total). Keep in mind here that you can add helper methods to your class to make the code cleaner and easier to debug.

Part D - Coding Style and Documentation (4 points)

This grade is awarded for proper coding styles. This includes:

- Appropriate prompts and informative output
- Appropriate method, variable and object names
- Proper indentation
- Proper java documentation
- Good commenting (explains what code is doing)
- Well-organized, elegant solution

Submission Requirements:

1. Submit your project export (**lastNameFirstNameProject2.zip**) that includes your program to fulfill the requirements.
2. Submit a PDF document with the name **project2Info.pdf** that includes the following assignment information:
    1. Explanation of any special or additional features you added, if any.
    2. If your solution to this problem is incomplete, you may explain the status as of your submission and describe any issues that kept you from completing the assignment.
    3. Discuss the easy and challenging parts of the assignment. How did you overcome all or some of the challenges?
    4. Did you learn anything that will help you perform better on future assignments?