Bookstore System Design & Implementation Assignment

Instructions: Please answer the following questions based on the provided lecture documentation.

Section 1: True / False (1 point each)

Indicate whether the following statements are true or false.

1. The `CashTill` class can only process objects of type `Book` and `Magazine`.

Answer: False

2. The `Editable` class is an interface that defines the contracts for `edit()` and `initialize()`.

Answer: False

3. The `Ticket` class inherits its properties and methods from the `Publication` abstract class.

Answer: False

4. When a `Book`'s `sellItem()` method is called, it decrements the `copies` count if there is stock available.

Answer: True

5. The system exclusively uses the older `java.util.Date` class for handling the `currentIssue` of a magazine.

Answer: False

Section 2: Multiple Choice (2 points each)

Choose the single best answer for each question.

6. What is the primary purpose of the `SaleableItem` interface in this system? a) To handle user input for item properties. b) To define a common contract for any item that can be sold. c) To store common properties for publications like title and price. d) To allow objects to be converted into a byte stream for storage.

Answer: B. To define a common contract for any item that can be sold

7. Which class is directly responsible for maintaining the `runningTotal` of all sales? a) `SaleableItem` b) `Publication` c) `CashTill` d) `ShopDemo`

   Answer: C. CashTill

8. The `CashTill.sellItem(SaleableItem item)` method's ability to work with `Book`, `Magazine`, and `Ticket` objects is a direct example of which Object-Oriented principle? a) Inheritance b) Encapsulation c) Polymorphism d) Serialization

   Answer: C. Polymorphism

9. According to the documentation, what is the correct string format for parsing the `currentIssue` date in the `Magazine` class? a) "yyyy-MM-dd" b) "dd/MM/yyyy" c) "MM-dd-yyyy" d) "dd-MM-yyyy"

Answer: D. “dd-MM-yyyy"

10. Which of the following is NOT a direct property of the `Publication` abstract class? a) title b) price c) author d) copies

Answer: C. author

11. What happens in one of the `Editable` class's `getInput()` methods if the user provides empty input (i.e., just presses Enter)? a) The method throws an `InputMismatchException`. b) The method returns `null` or zero. c) The method returns the default value that was passed into it. d) The method prompts the user to enter a value again.

Answer: C. The method returns the default value that was passed into it.

12. Based on the class hierarchy described, which of the following classes implements `SaleableItem` but does NOT extend `Editable`? a) `Book` b) `Magazine` c) `DiscMag` d) `Ticket`

Answer: D. Ticket

Section 3: Short Answer (3 points each)

Provide a brief, direct answer to the following questions.

13. What are the two abstract methods that any concrete subclass of `Editable` must implement?

Answer: The two abstracts methods are edit() and initialize()

14. What is the stated purpose of implementing the `Serializable` interface on classes like `Publication` and `Ticket`?

Answer: The purpose is to allow object of these types to be converted into a byte stream, which is useful for saving their state to a file or sending them over a network.

15. When implementing the `edit()` method in the `DiscMag` class, how does it handle the editing of properties it inherits from the `Magazine` class?

Answer: It calls the parent Magazine's method using super.edit() to handle the inherited properties first, and then adds its own logic for the hasDisc property.

16. Besides adding the item's price to the `runningTotal`, what other crucial action does the `CashTill`'s `sellItem` method perform on the item object it receives?

Answer: It calls the sellItem90 method on the item object irself, which triggers the specific selling logic for that item type (e.g. decrementing stock for a book.

17. What is the name of the class that serves as the test bed or demonstration for the entire system?

Answer: The ShopDemo class

18. What modern Java class is used to manage the `currentIssue` date property in the `Magazine` and `DiscMag` classes?

Answer: java.time.localDate

Section 4: Long Answer (6 points each)

Provide a detailed, well-structured answer to the following questions.

19. Explain in detail how the principle of **Polymorphism** is demonstrated by the `CashTill.sellItem(SaleableItem item)` method. Describe the flow of execution when a `Book` is sold versus when a `Ticket` is sold, highlighting how the `CashTill`

class can handle both uniformly without needing to know their specific types at compile time.

Answer: Polymorphism in this case is demonstrated by the CashTill.sellItem() method which takes a single parameter with the type SaleableItem, which means CashTill doesn't need to know specifically what class it is as long as it implements SaleableItem interface. This includes all the subclasses of the class that implement the interface.

Execution flow (Book):

1. Book object is a subclass of Publication which implements SaleableItem interface, so it must provide concrete method of sellItem();
2. When CashTill calls item.sellItem(), it'll call the sellItem() method defined in Book class.
3. The sellItem() will be executed and decrements the copies count and call getPrice() method to add to the runningTotal

Execution flow (Ticket):

1. Ticket object is a class that implements SaleableItem interface, so it must provide concrete method of sellItem();
2. When CashTill calls item.sellItem(), it'll call the sellItem() method defined in Ticket class.
3. The sellItem() will be executed and decrements the copies count and call getPrice() method to add to the runningTotal

20. Describe the distinct roles of the `SaleableItem` interface, the `Editable` abstract class, and the `Publication` abstract class in the application's design. Explain why the designers chose to separate these concerns and how this separation contributes to a more flexible and maintainable system.

Answer: The separate abstract classes and interface are useful to decouple the dependency of the code, so that they don’t need to know the details of each other to be used. This can improve flexibility and maintainability of the code.

- SaleableItem (Interface): its role is to define the contract for being sellable. It declares what’s the must-have behavior of a sellable item by defining sellItem() and getPrice(), which are the methods needed by CashTill to call sellItem to sell unrelated classes (Publication and Ticket).

- Editable (Abstract Class): its role is to provide a framework for interactive data entry and modification. It provides both abstract methods edit() and initialize() that calls force subclasses to implement this behavior, and concrete helper methods getInput() to reduce code duplication and increase consistency.

- Publication (Abstract Class): its role is to model the shared state and behavior of published media. It defines common attributes for publications (title, price, copies). As superclass, it provides default method implementations for its subclasses and make a good foundation for the “is-a” relationship.

  Contribution to Flexibility and Maintainability: The separation is very useful because using CashTill, it can use the sellItem() method without knowing the details of each SaleableItem. Editable() offers different methods to help creating new object for each item added to the database and reduce the needs to write the same getInput() method for each prompt. And Publication is useful to reduce the need for implementing and extending each class and interface needed. Publication creates a clear separation between each publication items such as Book, Magazine, etc with their own detail and method implementation and prevent unrelated objects into an inappropriate inheritance hierarchy.