**Assignment on Collection**

1. Create a **Book** class with the following properties:

**String title**

**String author**

**int year**

1. Generate getters and setters for all properties of the **Book** class using the **@Field** annotation.
2. Create a **Library** class with a single property:

* **List<Book> books**

1. Add a **addBook(Book book)** method to the **Library** class that adds a **Book** object to the **books** list.
2. Add a **removeBook(Book book)** method to the **Library** class that removes a **Book** object from the **books** list.
3. Add a **findBooksByAuthor(String author)** method to the **Library** class that returns a list of **Book** objects that have the specified author.
4. Add a **findBooksPublishedAfter(int year)** method to the **Library** class that returns a list of **Book** objects that were published after the specified year.
5. In the **Main** class, create a new **Library** object and add several **Book** objects to its **books** list.
6. Use the **removeBook()** method to remove one of the **Book** objects from the **books** list.
7. Use the **findBooksByAuthor()** method to find all **Book** objects in the **books** list that were written by a specific author, and print them to the console.
8. Use the **findBooksPublishedAfter()** method to find all **Book** objects in the **books** list that were published after a specific year, and print them to the console.
9. Bonus: Add validation to the **addBook()** method to ensure that each **Book** object added to the **books** list has a non-empty **title**, **author**, and a positive **year** value. If any of these conditions are not met, throw an exception with a helpful error message.

This assignment will help you practice creating and working with collections in Groovy, as well as using methods and objects to manipulate and query collections.

**Detailed explanation of the assignment on Groovy collections:**

1. The **Book** class is a simple class that represents a book. It has three properties: **title**, **author**, and **year**. These properties are defined using the **@Field** annotation to generate getters and setters for each property.
2. The **Library** class represents a collection of **Book** objects. It has a single property called **books**, which is a list of **Book** objects.
3. The **addBook(Book book)** method adds a **Book** object to the **books** list. It takes a **Book** object as a parameter and adds it to the end of the **books** list.
4. The **removeBook(Book book)** method removes a **Book** object from the **books** list. It takes a **Book** object as a parameter and removes it from the **books** list if it exists.
5. The **findBooksByAuthor(String author)** method returns a list of **Book** objects that have the specified author. It takes an author name as a parameter and returns a list of all **Book** objects in the **books** list that have the same author.
6. The **findBooksPublishedAfter(int year)** method returns a list of **Book** objects that were published after the specified year. It takes a year value as a parameter and returns a list of all **Book** objects in the **books** list that have a **year** value greater than or equal to the specified year.
7. In the **Main** class, we create a new **Library** object and add several **Book** objects to its **books** list using the **addBook()** method.
8. We then use the **removeBook()** method to remove one of the **Book** objects from the **books** list.
9. We use the **findBooksByAuthor()** method to find all **Book** objects in the **books** list that were written by a specific author, and print them to the console.
10. We use the **findBooksPublishedAfter()** method to find all **Book** objects in the **books** list that were published after a specific year, and print them to the console.
11. Bonus: We add validation to the **addBook()** method to ensure that each **Book** object added to the **books** list has a non-empty **title**, **author**, and a positive **year** value. If any of these conditions are not met, an exception with a helpful error message is thrown.

This assignment provides a good opportunity to practice working with collections in Groovy. By creating a **Library** class that contains a list of **Book** objects, we can demonstrate how to manipulate and query collections using methods and objects. Additionally, the bonus task of adding validation to the **addBook()** method allows students to practice error handling and input validation, which are important skills in any programming language.

Top of Form

Bottom of Form