University of Windsor

SCHOOL OF COMPUTER SCIENCE

COMP-2120, Fall 2024
Object Oriented Programming Using Java
LAB-7
TOTAL MARKS: 10

LAND ACKNOWLEDGEMENT

The School of Computer Science at the University of Windsor *sits on the Traditional Territory of the Three Fires Confederacy of First Nations*. We acknowledge that this is the beginning of our journey to understanding the Significance *of the history of the Peoples of the Ojibway, the Odawa, and the Pottawatomie*.

SUBMISSION DEADLINE: NOV-22, 2024

GENERAL INFORMATION

Welcome to the Lab Session of COMP-2120 where you will learn object-oriented programming (OOP) using Java. Please, arrive early to the lab and get settled to start working on the lab. Arriving 20 minutes after the start of the lab will be considered late and can affect your lab participation point. There are eight lab exercises you need to complete throughout the course. Each week on Thursday morning we will publish the lab exercise. You need to upload your code by Thursday, 10 AM in the next week.

LAB GRADING AND ASSESSMENT

You need to explain your code to TAs in the lab to receive the marks. YOU MUST demonstrate your results and explain the code to the TAs to receive marks. Thus, attending the lab sessions is mandatory. Note that your TAs are here in the Lab to help you learn. Feel free to ask questions and talk to them. Remember, the total of eight Labs is worth 16% of your final grade.

Lab-7

Create a robust backend for a digital library system that handles various types of media items (e.g., books, movies, podcasts, and music albums). This lab will focus on using generics, inheritance, and the Collection Framework to build a flexible and type-safe data management system.

Description

Imagine you are setting up the backend of a personalized digital library system. Your library will store different types of media items and offer ways to organize, search, and filter the data.

Task 1: Set Up the Media Class Hierarchy **(2 points)**

1. **Create the Base Class Media**:
    - **Attributes**: title, genre, year, and rating (e.g., out of 5 stars).
    - **Methods**: Basic getters, setters, and a toString() method to display information.
2. **Define Media Subclasses**:
    - **Book**: Additional attributes include author and ISBN.
    - **Movie**: Additional attributes include director, duration, and MPAA rating.
    - **Podcast**: Additional attributes include host and episodeCount.
    - **MusicAlbum**: Additional attributes include artist, tracks, and releaseDate.

Each subclass should have a constructor and toString() method that provides a formatted output of all its attributes.

Task 2: Implement the DigitalLibrary<T extends Media> Class Using Generics **(3 points)**

The DigitalLibrary<T> class will manage a collection of media items of type T. Use the Collection Framework to handle the data and generics to ensure type safety.

Key Methods to Implement:

1. void addMedia(T item): Adds a media item to the collection.
2. boolean removeMedia(T item): Removes a media item from the collection.
3. T searchByTitle(String title): Searches for a media item by title.
4. List<T> filterByGenre(String genre): Returns items matching a genre.
5. List<T> recommendByRating(int minRating): Returns items with a rating above a specified threshold.

Requirements:

- Use a List (e.g., ArrayList) to store media items.
- Ensure type safety by leveraging generics.
- Implement a custom exception (ItemNotFoundException) for cases where an item is not found. (**Optional/ You can may be display an error/not found message instead**)

Task 3: Implement Sorting and Filtering **(3 points)**

1. **Sorting**:
    - Use Comparator to implement custom sorting for media items by title, year, and rating.
    - Add methods like sortByTitle() and sortByYear() to the DigitalLibrary class.
2. **Filtering**:
    - Implement methods to filter media items by genre and other attributes as applicable (e.g., filter books by author, movies by director).

Task 4: Testing **(2 points)**

Create a DigitalLibraryTest class to test all methods:

- **Add and Remove Items**: Test adding and removing different media items.
- **Sorting**: Test sorting by title, year, and rating.
- **Filtering**: Test filters like genre-based filtering.
- **Error Message**: Verify that the ItemNotFoundException is correctly thrown (optional)/ Or correct message displayed for item not found.

WHAT DO YOU NEED TO DO?

1. Complete the program.
2. Attend the lab. Explain the code and the purpose of using different OOP features to a TA.
3. TAs can modify the input to your program.