PGR112 Object-oriented Programming

Re-sit Exam

2022

Home exam, individual

Notice:

- The main topics in this exam are Java, JDBC, and diverse Object-oriented programming theory as given through the course.
- All code relevant to the solution must be developed by you. Except for in very special and concrete cases, you can't copy paste code that is not yours. You can use the code in the course Git repository as reference. If you do use existing code, you must clearly show (in comments) what the source was.
- You should put all the files that the delivery contains in one folder and zip before uploading to WISEFlow.
- The solution should be able to run properly in IntelliJ (i.e. not have to run with any other special technology/IDE)

If you are unsure about something regarding the exam, re-read the instructions and try to make a best assumption. Clearly express what assumptions you made and why in the report (see below).

The delivery will contain three things:

1. The project folder with Java, folders, txt or json files if any, etc.
2. The database schema files you have created.
3. A small report (pdf or word document) that shortly describes how you have implemented the major OOP concepts through your design.

Book Loan System

You are given the task to implement an interactive book loan system for a library.

In the book loan system, there are two kinds of books: normal books, and acoustic books (lydbok). You must have a certain number of hard copies for each normal book, and the number of hard copies is up to you to decide. For acoustic books, you need to subscribe for the digital copy. The digital subscription is unlimited, but there is a free trial period for each acoustic book that is up to you to decide.

You shall create all books and store them in database. You shall design an interactive user interface, which reads user inputs, communicates with database, and gives response to users. The major interactive activity is for a user to loan and return books. Note that a user can at most keep one hard copy or subscribe one digital copy, depending on the book type, but it is allowed to loan the same book after return. It shall be easy for the user to quit the book loan system at any time.

You need to sign up as a user in order to use the book loan system.

The book loan system shall keep track of loan records, i.e., which book is loaned by which user. The loan records are stored in database, and shall facilitate insert, delete and search operations etc.

In addition, you shall make a report (200-400 words) in a pdf or word document about the following things:

- What is Objected-oriented programming? What concepts did you use in your solution? How did you use these concepts to develop your solution?
- What is JDBC? How did you use JDBC?
- Examples of OOP concepts that you have implemented in your Book Loan system design.

What are you to make?

- **Database and tables**
    - You will create a database called **bookDb**.
    - You will create two tables that stores normal books and acoustic books respectively. One table is called **normalBook**, the second table is called **acousticBook**. You shall insert book information into tables using JDBC connection.
        - The **normalBook** table contains following columns for each record: **id; author; year; language; numberofHardCopies; loanPeriod**.
        - The **acousticBook** table contains following columns for each record: **id; author, language; freeTrialPeriod**.

      The tables shall be updated based on the user activity. For example, if a user loans a normal book, the numberofHardCopies will be decreased by 1. If a user returns a normal book, the numberofHardCopies will be increased by 1. Note that a user cannot loan a book with no copies available in the system.
    - You will create a table called **loanRecord** that stores information for each loan activity. You shall insert a new loan record if a user loans a book; delete the loan record if the user returns a book.
        - The **loanRecord** table shall at least contain following columns for each record: **id; user; bookType; bookId.** Note that bookId is related to id in normalBook and acousticBook.
    - Other things you think should be included.
- **Interactive Java program**
    - You shall use Java.Util.Scanner to build the interactive Book Loan system.
    - You shall be able to use the book loan service for unlimited times unless you want to quit the system.
    - You shall be able to choose the books you would like to loan.
    - You can only return the books you have loaned.
    - You shall be able to do different queries towards database, i.e., query which books are loaned by a given user.
    - The interactive GUI shall display necessary information to facilitate user operation, i.e., numberofHardCopies etc.

As an example, your Book Loan system terminal could be about the following steps:

1. Sign up a new user:
   Enter user1 at terminal
2. Choose your activity:
    A. Loan a normal book
    B. Loan an acoustic book
    C. Return a normal book
    D. Return an acoustic book
    E. Check loan records
    F. Exit

Choose A for Loan a normal book

3. Below books are available. Choose the one you would like to loan:
   A. Snowman Jo Nesbo 2007 Norwegian 2
   B. Magpie Murders Anthony Horowitz 2016 English 3

   Choose A
4. You have successfully loaned Snowman! Would you like to
   A. Loan a normal book
   B. Loan an acoustic book
   C. Return a normal book
   D. Return an acoustic book
   E. Check loan records
   F. Exit

   Choose D, return an acoustic book
5. Error! You have not loaned any acoustic book. Would you like to:
   A. Loan a normal book
   B. Loan an acoustic book
   C. Return a normal book
   D. Return an acoustic book
   E. Check loan records
   F. Exit

   Choose C, return a normal book

6. You have loaned below books. Choose the one you would like to return:
   A. Snowman Jo Nesbo 2007 Norwegian 2

   Choose A
7. You have successfully returned Snowman! Would you like to:
   A. Loan a normal book
   B. Loan an acoustic book
   C. Return a normal book
   D. Return an acoustic book
   E. Check loan records
   F. Exit

   Choose E
8. Please enter the username: enter "user1"
9. The user1 does not have any loaned books! Would you like to:
   A. Loan a normal book
   B. Loan an acoustic book
   C. Return a normal book
   D. Return an acoustic book
   E. Check loan records
   F. Exit

   Choose F
10. Bye and welcome back!

Note that the example is just for your reference. You have a lot of freedom regarding the user interactive program and how to show information. You can add as much details and complexity as you can.

Other details

- You can create a **Book** class as an abstract class or interface. You also create **NormalBook** and **AcousticBook** classes as child classes of Book class. Note that NormalBook and AcousticBook have different attributes. And NormalBook and AcousticBook classes are connected to normalBook and acousticBook tables in database.
    - You shall design common functions in the Parent Book Class shared by child classes. For example showBook() – it will show current Book information. The function implementations are expected to be different in NormalBook and AcousticBook.
    - You shall design functions that are only available in current child class. For example, NormalBook class shall include a function called isHardCopyAvailable() or similar. If there are no available hard copies, the user shall be notified and not able to loan.

  Think about inheritance and abstraction concepts we have learned in this course.

- You can create a **UserLoan** class. A UserLoan class is initiated with a new object each time when you sign up as a user. The UserLoan class is used to keep tracks of loan activities of current user.

- You can create a **LoanRecord** class. The LoanRecord class is to update loan record for all users. It is connected to the loanRecord table in database.

- The way to communicate with database.
    - You shall insert all books into database using JDBC connections.
    - You shall be able to fetch available books of chosen type, i.e., normal or acoustic.
    - If a user loans a normal book, the numberofHardCopies will be decreased by 1. If a user returns a normal book, the numberofHardCopies will be increased by 1.
    - A user cannot loan a book with no copies available.
    - You shall insert a new loan record to loanRecord table if a user chooses to loan a book; delete the loan record if the user chooses to return a book.
    - You shall be able to do queries towards database i.e., fetch books loaned by a give user.
    - Feel free to include more interactions between user and database.

Assessment criteria, for students and assessors

The following will be the main points to assessed:

- Java, JDBC, and Object-oriented programming is what is assessed in this exam.
- The basic OOP concepts such as Inheritance, Polymorphism, Abstraction and Encapsulation.
- Data types such as ArrayList, HashMap, etc.
- Try to use advanced Java techniques such as Iterators, Optional and Lambda expressions.
- Exception handler to handle exceptions, and input validation to validate user input.
- Amount of (good) code and complexity are of importance
- Folder structure, tidy code, naming convention

Guideline for how much each part counts

The percentage numbers below present approximate numbers on how much each part will count

- User Interface logic: ca. 20%
- Use of OOP concepts: ca. 20%
- Use of JDBC: ca 20%
- Java techniques (data type, advanced techniques, exception handler etc): ca. 15%
- Report: ca. 10%
- Code structure, tidy code, good names on variables and functions, etc.: ca. 10%
- *Amount of code and complexity: ca. 5%

*All points affect each other. The assessment will require a holistic view of techniques applied in the exam delivery.

--End of exam text--