PROGRAMMING ASSIGNMENT 1

TAs : Alpay TEKİN
Due Date : 30.03.2025 (23:00:00)
Programming Language: Java 8 (Oracle)

University Library Management System

1 Introduction

Object-Oriented Programming (OOP) is a fundamental paradigm in software development that enhances code reusability, scalability, and maintainability through principles like encapsulation, inheritance, and modular design. Java, as a widely adopted OOP language, provides a robust and secure environment for building complex applications, making it an ideal choice for developing structured and efficient software solutions.

In this assignment, you will implement a University Library Management System using OOP principles, specifically focusing on inheritance and encapsulation. This project will simulate user roles such as students, academic members, and guests, each with distinct borrowing rules, ensuring a realistic and well-structured system.

2 System Definition

In this part, you will find the basic attributes of the users and items. You must implement at least those attributes.

2.1 Library Items

There are 3 types of library items: book, magazine, and DVD. Items can be borrowed and returned to the library by users.You must consider followings:

- Each item has **unique** ID, name and type.

- Type can be **normal**, **referenced**, **rare** or **limited**.
- Each book has author, genre.
- Each magazine has publisher and category.
- Each DVD has directory,category and runtime. Runtime indicates the duration of the films in minutes.

2.2 Users

There are 3 types of users: student, academic staff and guest user. Each can borrow and return library items based on specific rules. Each user has **unique** ID, name and phone number.

- Student has faculty, department and grade.
- Academic staff has faculty, department and title.
- Guest user has occupation.
- Each user type must follows certain rules to borrow item. These rules are presented in table 1.
- **Max Items** refers to maximum number of item that each user can borrow at the same time.
- If a user does not return an item by the overdue date, the system will automatically return the item to the library and apply a 2$ penalty to the user's account.
- If user's penalty amount equal or greater than 6$, user must pay the penalty amount before borrowing new item.

User Type	**Max Items**	**Overdue (days)**	**Penalty Threshold**	**Restricted Items**
Student	5	30	Cannot borrow if penalty ≥ 6	Cannot borrow Reference Items
Academic Member	3	15	Cannot borrow if penalty ≥ 6	No restrictions
Guest	1	7	Cannot borrow if penalty ≥ 6	Cannot borrow Rare or Limited Items

Table 1: Borrowing Rules for Different User Types in the Library System

3 Definition of Input and Output

The program will read input data from text files, including user information, library items, and borrowing requests. After processing the requests, the system will write the results to an output text file, recording all transactions, borrowing statuses, and penalties applied.

3.1 Inputs

Items will be read from item.txt file. It contains, item class (B for book, M for magazine and D for DVD) and other required information about the item as shown in the below example.The input format for library items varies based on the type of item. Each entry follows a structured format:

- For books, <Item Class>,<ID>,<Title>,<Author>,<Category>,<Type>.
- For magazine, <Item Class>,<ID>,<Title>,<Publisher>,<Category><Type>.
- For DVD, <Item Class>,<ID>,<Title>,<Director>,<Category>,<Runtime>,<Type>.

```
B,1001,The Great Gatsby,F. Scott Fitzgerald,Classic Fiction,normal
B,1002,Dune,Frank Herbert,Science Fiction,normal
B,1003,Pride and Prejudice,Jane Austen,Romance,normal
B,1004,The Silent Patient,Alex Michaelides,Thriller,normal
B,1005,Sapiens: A Brief History of Humankind,Yuval Noah Harari,Non-
   Fiction,reference
B,1006,The Hobbit,J.R.R. Tolkien,Fantasy,rare
B,1007,Gone Girl,Gillian Flynn,Mystery,limited
B,1008,The Alchemist,Paulo Coelho,Philosophical Fiction,reference
M,2001,National Geographic,National Geographic Society,Science,normal
M,2002,Time,Time USA,News,normal
M,2003,Scientific American,Springer Nature,Science,reference
M,2004,Vogue,Conde Nast,Fashion,normal
2005,The Economist,The Economist Group,Economics,limited,
D,3001,The Shawshank Redemption,Frank Darabont,Drama,142 min,normal
D,3002,The Godfather,Francis Ford Coppola,Crime,175 min,normal
D,3003,Inception,Christopher Nolan,Sci-Fi,148 min,limited
D,3004,Pulp Fiction,Quentin Tarantino,Crime,154 min,normal
D,3005,The Matrix,Wachowski Sisters,Sci-Fi,136 min,rare
```

Table 2: Example Input Format for Library Items

Users will be read from users.txt file. It contains, item class (S for student, A for academic member and G for guest user) and other required information about the users as shown in the below example.The input format for users varies based on the type of item. Each entry follows a structured format:

- For student, <class_type>,<name>,<ID>,<phone_number>,<department>,<faculty>,<grade>
- For academic member, <class_type>,<name>,<ID>,<phone_number>,<department>,<faculty>,<t
- For guest user, <class_type>,<name>,<ID>,<phone_number>,<occupation>

User actions will be read from commands.txt.You can find the input structure for different action below:

- For borrow operation, <borrow>,<userID>,<itemID><borrowDate>.
- For return operation, <return>,<userID>,<itemID>.

```
S,John Smith,1001,555-1234,Computer Science,Engineering,1
S,Emily Johnson,1002,555-2345,Electrical Engineering,Engineering,3
S,Michael Brown,1003,555-3456,Biology,Science,2
A,Dr. Sarah Wilson,2001,555-4567,Physics,Science,Professor
A,Dr. Robert Thompson,2002,555-5678,Literature,Arts,Associate Professor
A,Dr. Laura Martinez,2003,555-6789,Mathematics,Science,Assistant
   Professor
G,David Lee,3001,555-7890,Writer
G,Jennifer Parker,3002,555-8901,Researcher
```

Table 3: Example Input Format for Users

- displayUsers command will write information about each user(sorted ascending based on user ID).
- displayItems commands will write information about each items (sorted ascending based on item ID).
- pay operation: pay,<userID>

```
borrow,1001,1001,10/03/2025
borrow,1001,1002,10/03/2025
borrow,1001,1003,10/03/2025
borrow,1001,1004,10/03/2025
borrow,1001,2001,10/03/2025
borrow,1001,1006,10/03/2025
return,1001,2001
borrow,1001,1006,10/03/2025
displayUsers
displayItems
```

Table 4: Example Input Format for Commands

3.2 Output

The program will write the output to a file, and the filename will be provided as a command-line argument. Users must strictly follow the specified file format, as the correctness of the output will be verified using the diff command. More sample inputs and outputs will be shared on Piazza.

4 Helper Functions

4.1 Reading Text Document

To read the input values from text file, you can utilize from following function.

```
<return_type> readTxt(String input){
    try (BufferedReader br = new BufferedReader(new FileReader(
       input))) {
        String line;
        while((line = br.readLine()) != null){
            String[] parts = line.split(",");

            // variable parts stores the each value in row,
               separated by comma
            //you can access the each values in a row using
               array index
            // implement the logic
        }
    }catch (IOException e) {
        e.printStackTrace();
    }
    return statement;
}
```

5 Test and Execution

Your code must be compiled and executed under Java 8 and dev.cs.hacettepe.edu.tr. If your code does not compile and execute under developer server, then you will be graded as 0 for code part even if it works on your own machine. Your program must take input files,and name of the output file. Sample run command is as follows:

```
javac Main.java
java Main items.txt users.txt commands.txt output.txt
```

6 Grading Policy

Task	Grade
Class Implementations using OOP Principles	45(5 per each, 5 full implementation)
Correct Output	45*
Comments in JavaDoc Style	10**
Total	100

***Even though producing correct output and commenting seems like enough to get full credit, you must obey to the given rules in the PDF (for example using concept of OOP and four pillars of OOP etc.) otherwise you may face with some point deductions which may result with a grade that is as low as zero. There will**

be two overall multipliers about quality of your OOP and clean code separately which will vary in between 0 and 1! Note that there may be any other multipliers or point deductions in case of violation of the rules.

**** The score of the clean code comment part will be multiplied by your overall score (excluding clean code comment part) and divided by the maximum score that can be taken from these parts. Say that you got 45 from all parts excluding clean code comment part and 10 from clean code comment part, your score for clean code comment part is going to be 10*(45/90) which is 5 and your overall score will be 45+5=50.**

7 Submit Format

File hierarchy must be zipped before submitted (Not .rar, only .zip files are supported by the system).

```
-b<studentID>.zip
  -Main.java, or *.java
```

Late Policy

You have two days for late submission. You will lose 10 points from maximum evaluation score for each day (your submitted study will be evaluated over 90 and 80 for each late submission day). You must submit your solution in at the most two days later than submission date, otherwise it will not be evaluated. Please do not e-mail to me even if you miss the deadline for a few seconds due to your own fault as it would be unfair for your friends, e-mail submissions will not be considered if you do not have a valid issue.

Notes and Restrictions

- Your code must be able to execute on our department's developer server (dev.cs.hacettepe.edu.tr).
- You must use JavaDoc commenting style for this project, and you must give brief information about the challenging parts of your code, do not over comment as it is against clean code approach. Design your comments so that if someone wants to read your code they should be able to easily understand what is going on. You can check here to access Oracle's own guide about JavaDoc Sytle.
- **Use inheritance,and encapsulation where it makes logical and practical sense. Otherwise, you risk losing points for not properly applying OOP principles.**
- You must obey given submit hierarchy and get score (1 point) from the submit system.
- Do not miss the submission deadline.
- You can benefit from Internet sources for inspiration but do not use any code that does not belong to you.
- You can discuss high-level (design) problems with your friends but do not share any code or implementation with anybody.

- Source code readability is a great of importance. Thus, write READABLE SOURCE CODE, comments, and clear MAIN function. This expectation will be graded as "clean code".
- Use UNDERSTANDABLE names for your variables, classes, and functions regardless of the length. The names of classes, attributes and methods must obey to the Java naming convention. This expectation will be graded as "coding standards".
- You can ask your questions through course's Piazza group, and you are supposed to be aware of everything discussed in the Piazza group. General discussion of the problem is allowed, but **DO NOT SHARE** answers, algorithms, source codes and reports.
- All assignments must be original, individual work. Duplicate or very similar assignments are both going to be considered as cheating.