CSCI3170 (2015-2016 2nd term)
Introduction to Database Systems
Project– Library Inquiry System

Group Registration Deadline: 23:59 15th February 2016
Phase 1 Deadline: 23:59 10th March 2016
Phase 2 Deadline: 23:59 17th April 2016

1. Introduction

You are required to implement a Library Inquiry System for a library so that all information about books in the library, library users, and loans are stored. The system has to support interactive operations from operators. You are required to use Java JDBC API to access the database. Our tutors will give a tutorial on how to use the JDBC API. You are required to implement a Java command line application to realize all system functions stated in this specification.

There are two phases. In phase 1, you are required to design the database for the system (including an ER-diagram and a relational schema). After the deadline of Phase 1, a suggested solution will be provided. You are required to use the suggested solution to complete Phase 2. In Phase 2, you are required to implement the Library Inquiry System as a Java command-line program. Our tutors will give tutorials on how to connect to a database system and deploy your work on the required platform.

This is a group project and each group consists of three members. ONLY one copy of solution is required for each group. Please sign the group registration on our course homepage before the group registration deadline.

2. Milestones

Preparation

- Read the document thoroughly and make sure you understand all the assumptions and regulations stated in Section 4.

Phase 1 (20 %)

- According to the data specifications in Section 3, design an ER-diagram and transform it into a relational schema without any redundant fields and tables.

Phase 2 (80 %)

- According to the suggested solution of Phase 1, implement a Java application that fulfills all requirements stated in Section 5.
- Debug your system with different datasets and user inputs.
- Write a readme file to describe the compilation and deployment of your system.

3. Data Specification

All data files of the system are in Linux text file format (i.e. Newline character is **\n**) encoded in ASCII. Your Java command line application is required to read records stored in the files and inserts them into appropriate tables of the provided Oracle DBMS via JDBC API. There are four input files, a list of categories,

a list of library users, a list of books and a list of check-out records. Each line of each input file is a sequence of attributes delimited by tab (**\t**) character. The definition of each attribute in each input file is defined in the corresponding subsections. On the other hand, the ordering of the attributes within a line of each input file is the same as the ordering of the attribute in the corresponding sub section. A sample data set will be provided after the deadline of Phase 1.

3.1. Categories

Library users are divided into different categories. The loan period and the maximum number of books that can be borrowed by a library user are determined by his or her category. Each category has a unique category ID.

Item Name	Format	Description
Category ID	Non-empty positive integer with 1 digit	A unique identifier for a category.
Max books	Non-empty positive integer with at most 2 digits	The maximum number of books that can be borrowed by the library user in the corresponding category.
Loan period	Non-empty positive integer with at most 2 digits	The maximum number of days for the library user in the corresponding category to borrow a book.

3.2. Library Users

Each library user has a unique user ID and belongs to exactly one category.

Item Name	Format	Description
User ID	Non-empty string with 10 characters	A unique identifier of the library user.
Name	Non-empty string with at most 25 characters	The name of the library user.
Address	Non-empty string with at most 100 characters	The address of the library user.
Category ID	Non-empty positive integer with 1 digit	It indicates to which category the library user belongs.

3.3. Books

Each book is a literary composition written by one or more authors and a library stores a number of physical copies for each book. Each book has a unique call number

Item Name	Format	Description
Call number	Non-empty string with 8 characters	It is used for the library users to search for the book.
Number of copies	Non-empty positive integer with 1 digit	The number of identical copies of the book.
Title	Non-empty string with at most 30 characters	The title of the book.
Author(s)	Non-empty string with at most 25 characters for each author	Author name(s) of the book concatenated as a string with comma character as the delimiter.
Date of publication	Non-empty date in the format of DD/MM/YYYY	The date that the book is published

3.4. Checked-Out Records

Each check-out record shows a borrow history of a library user.

Item Name	Format	Description
Call number	Non-empty 8 characters	The call number of the checked-out book copy.
Copy number	Non-empty 1 digit Positive integer	The copy number of the checked-out book copy.
User ID	Non-empty 10 characters	The user ID of the borrower.
Check-out date	Non-empty date in the format of DD/MM/YYYY	The date that the book is checked-out.
Return date	Date in the format of DD/MM/YYYY	The date that the book is returned.

4. Assumption and Regulations

4.1. System

- All numerical values will not be larger than the maximum integer value that can be handled by Java.
- The system is case sensitive.
- All dates should have the following format if expressed in string: [DD]/[MM]/[YYYY] and have the same time zone as Hong Kong. (Note: Y=year, M=month, D=day)
- There is no duplicate row in any input and output.
- Your Java program may assume that any value entered into any input field is **correct in format only**.
- Your Java program may assume that all input data files are **correct in format and content**.

4.2. Categories

- Each category has a unique *category id* and it can be used to identify a category.
- Some categories may have the same *max books* or *loan period*.

4.3. Library Users

- Each library user has a unique *user ID* and it can be used to identify a library user.
- Some library users may have the same *name* or *address*.
- A user can only belong to one *category*.

4.4. Books

- Each book has a unique *call number* and it can be used to identify a book.
- Each book must have at least one *author*.
- Some books may have the same *title* or *author*.

4.5. Book Copies

- A book copy refers to a physically existing book copy that can be borrowed by a library user in the library.
- Each book copy of a book has a *copy number*. If a book has n book copies, the *copy number* of those n book copies ranges from 1 to n. For example, if a book has four copies, these four book copies have the same *call number*, and their *copy numbers* are 1, 2, 3 and 4.
- Each book copy has a unique pair of *call number* and *copy number* and they can be used jointly to identify a book copy.

4.6. Authors

- An author can be uniquely identified by his/her *name*.
- An author may write more than one book.

4.7. Checked-Out Records

- Each check-out record has a unique set of {*Call number*, *Copy number*, *User ID* and *Check-out date*} and this set of attributes can be used to identify a check-out record.
- Some library users may have never checked out any book copies.
- Some book copies may have never been checked out.
- A user may borrow the same book copy for more than one time.
- The *return date* of a book copy is NULL if the book copy is not returned.
- For each book copy, there is only one check-out record with a NULL return date.

5. System Function Requirements

You are required to write a simple command line application in Java. After performing a function, the program should display the last appeared menu. The Java program should consists of the following functions:

5.1. Administrator

The functions that can be used by an administrator are:

- **Create table schemas in the database**: This function creates all the tables for this system based on the relational schema given.

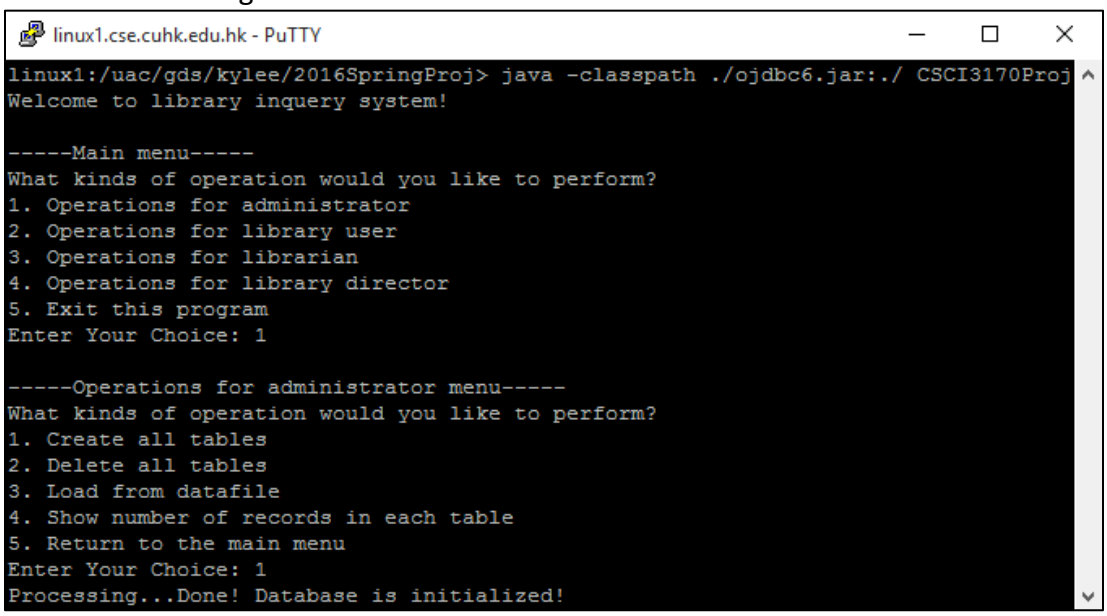

Figure 1: Expected interactive input and output while creating table schemas in Oracle DBMS.

- **Delete table schemas in the database**: This function deletes all existing tables in the system.

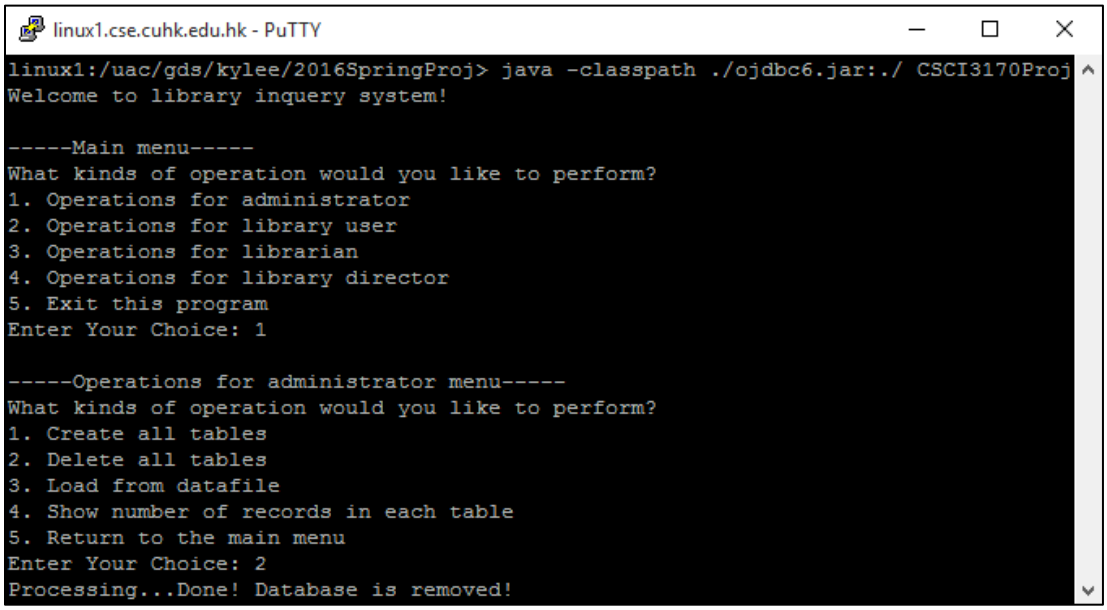

Figure 2: Expected interactive input and output while deleting table schemas from Oracle DBMS.

- **Load data from a dataset**: After a user enters the path of the folder that contains the data files, the system reads all data files from the user-specified folder and inserts the records into the appropriate table in the database. (Your program can assume that the user-specified folder must contain all 4 data files. These 4 input files are named category.txt, user.txt, book.txt and check_out.txt. Each data file stores the data corresponding to its filename.)

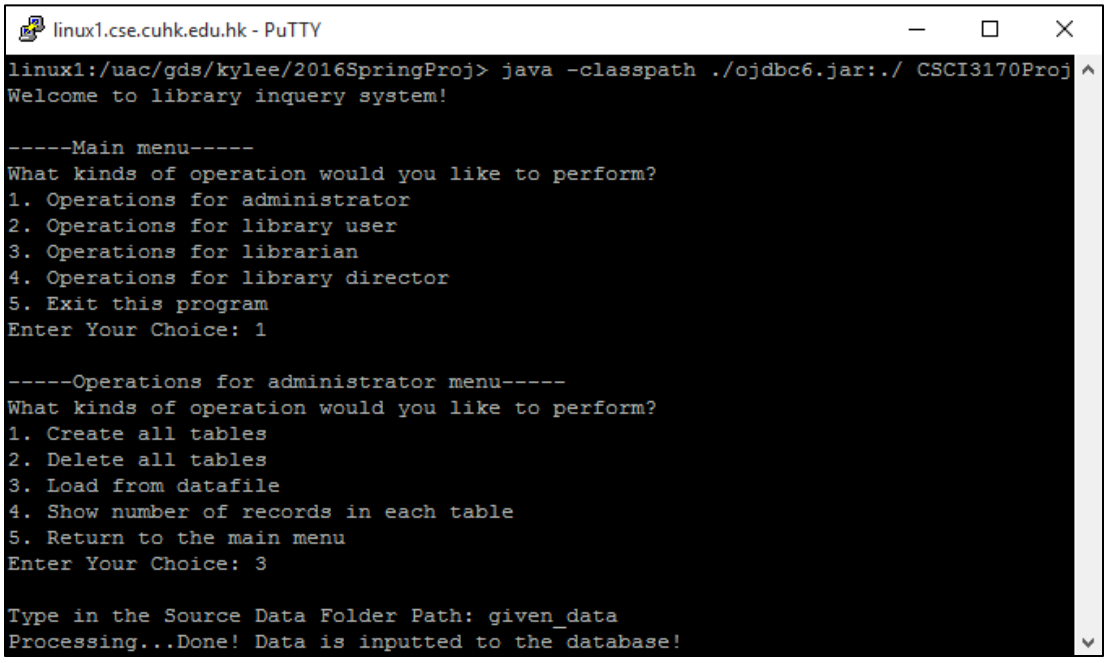

Figure 3: Expected interactive input and output while loading table schemas from the database

- **Show the number of records in each table:** This function shows the total number of records in each existing table.

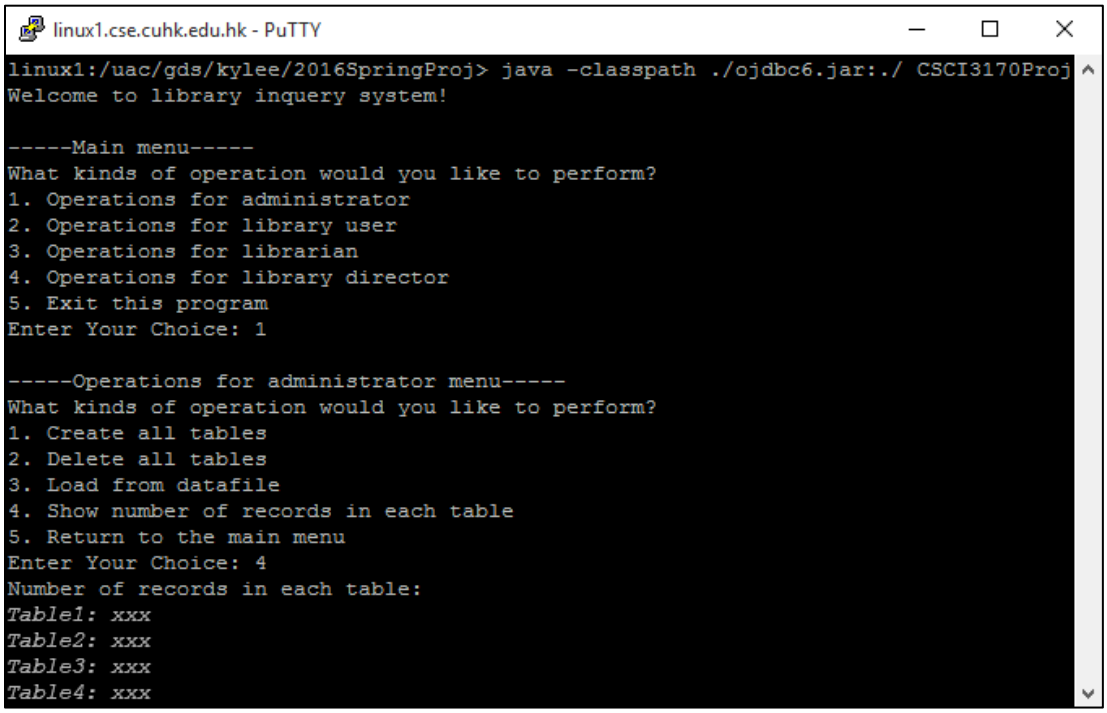

Figure 4: Expected interactive input and output while showing number of records in each table.

Note: Please replace words in Italic (i.e. *Table1: XXX, Table2: XXX, Table3: XXX, Table4: XXX*) in figure 4 with the tables in relational schema given in the suggested solution of phase 1. The number of tables may not be four as shown in Figure 4.

5.2. Library User

- **Search for books**: The system has to provide an interface to allow a library user to search for the books in the library in three different ways
  - By call number (exact matching)
  - By title (partial matching)
  - By author (partial matching)

You can assume that only one searching method can be selected by the library user for each query and the whole string entered by the library user is considered as one search word (e.g. When a user entered “Database Concept”, The system will consider “Database Concept” as one and only one search keyword instead of two search keywords “Database” and “Concept”). After the library user entered the search keyword, the program should perform the query and return all matching books in terms of their *call number*, *title*, *authors* and *number of available copies*. The results of the query should be sorted in ascending order of *call number* and outputted as a table as follows:

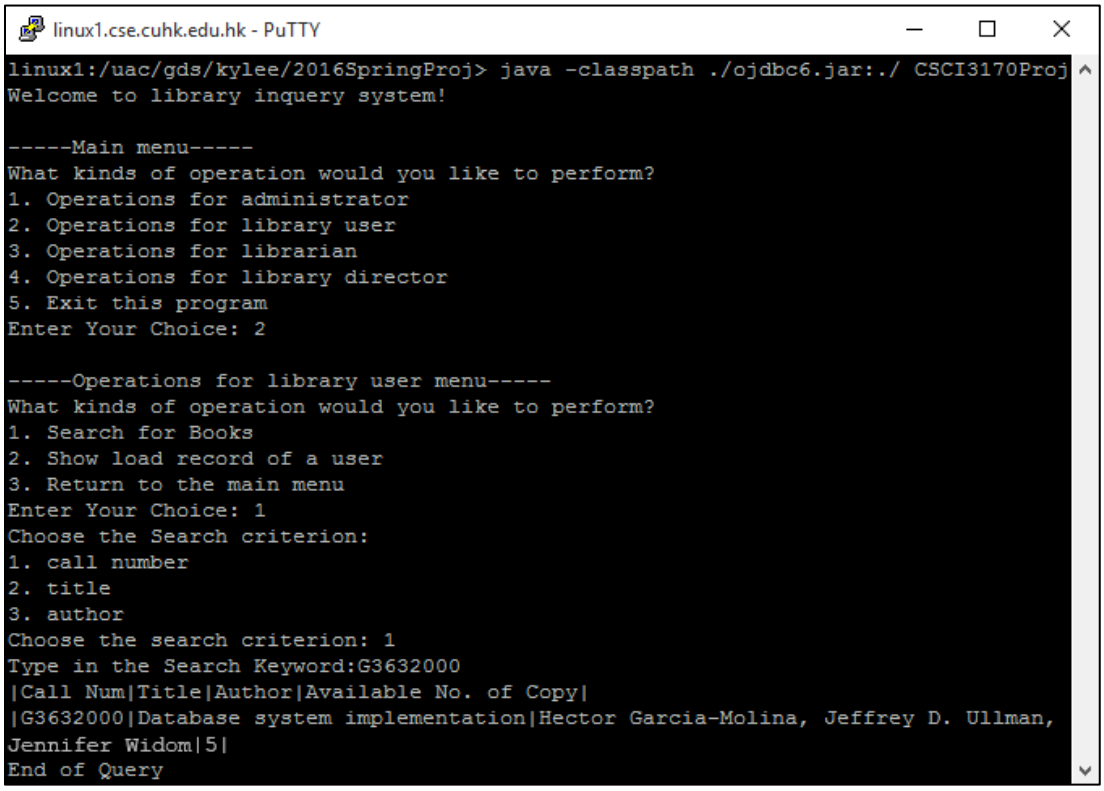

Figure 5: Expected input and output while searching for books

- **Show all check-out records of a library user:** The system has to provide an interface to allow a library user to show all his/her check-out records of with a given *user ID*. After the library user enters his/her *user ID*, the program will perform the query and return all the matching check-out records in terms of *call number*, *copy number*, *title*, *authors*, *check out date* and *whether the book copy of the corresponding check-out record is returned*. The check-out records should be sorted in descending order of *check-out date* and outputted as a table as follows:

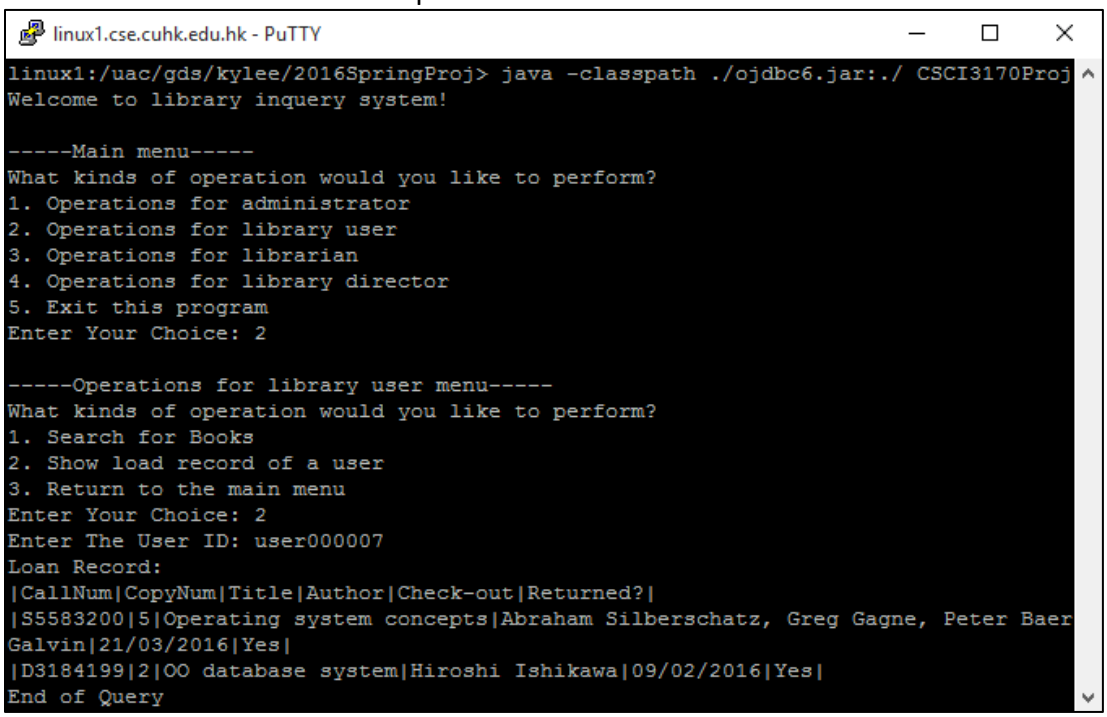

Figure 6: Expected input and output while showing all check-out records of a library user

5.3. Librarian

- **Borrow a book copy:** A librarian can perform the book borrowing procedure through the Library Inquiry System. First, he/she needs to input *call number* and *copy number* of the book copy being borrowed and the *user ID* of the library user. Then the system should check whether that book copy is available to be borrowed (i.e. There is no check out record of the specified book copy with NULL *return date*). If the book copy is available, it is then borrowed and a new check-out record of the specified book copy and user with NULL *return date* should be added to the database accordingly. Finally there should be an informative message whether the book copy can be lent successfully in layman terms.

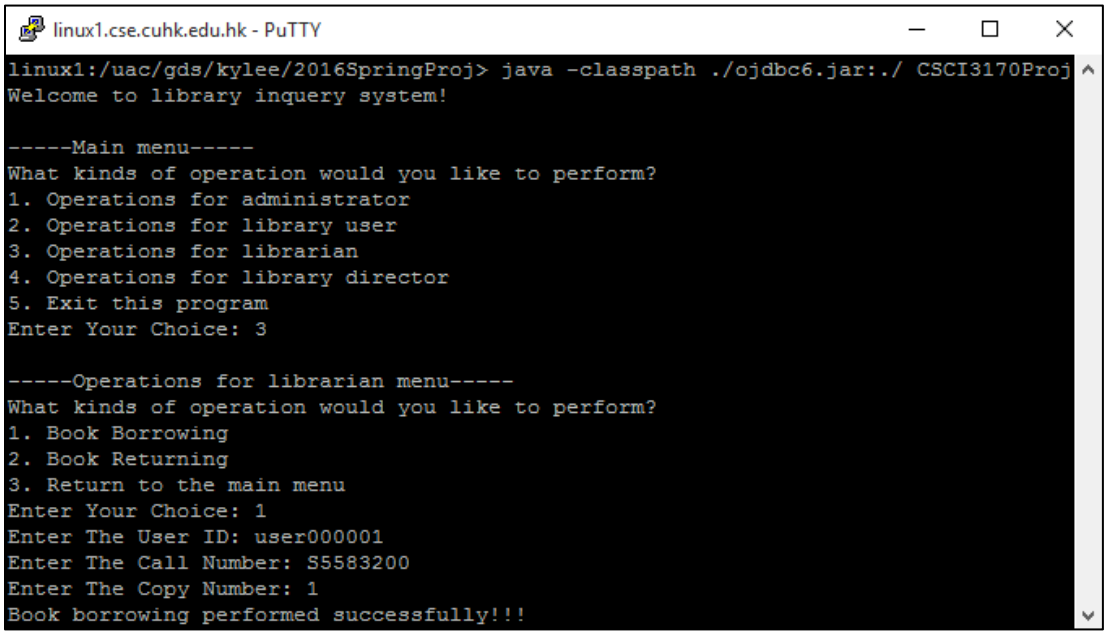

Figure 7: Expected input and output while a librarian process a book borrowing request

- **Return a book copy:** A librarian can perform the book returning procedure through the Library Inquiry System. First, he/she needs to input *call number* and *copy number* of the book copy being borrowed and the *user ID* of the library user. Then the system should check if a check-out record corresponding to the specified *user ID*, *call number* and *copy number* exists. If such record is found, the book copy can be returned and the *return date* of the check-out record found is updated to be the current date of the database server. Finally there should be an informative message whether the book copy can be returned successfully in layman terms. (For the sake of simplicity, you are not require to check whether the book is overdue or not)

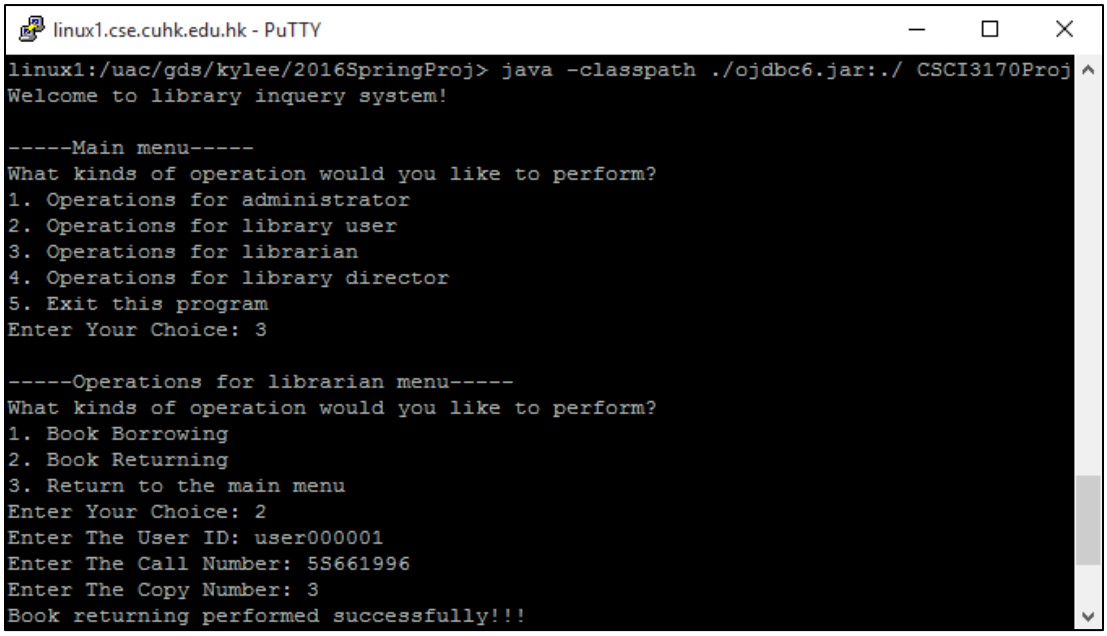

Figure 8: Expected input and output while a librarian process a book returning request

5.4. Library director

- **List all un-returned book copies which are checked-out within a period:** The system has to provide an interface to allow a library director to list all un-returned book copies which are checked-out within a given period (e.g. from 16/01/2015 to 15/02/2015). After the library director enters the period, the program will perform the query and return a list of all overdue book copies in terms of *user ID, call number*, *copy number and check-out date* in descending order of *check-out date* within the inputted period *inclusively*.

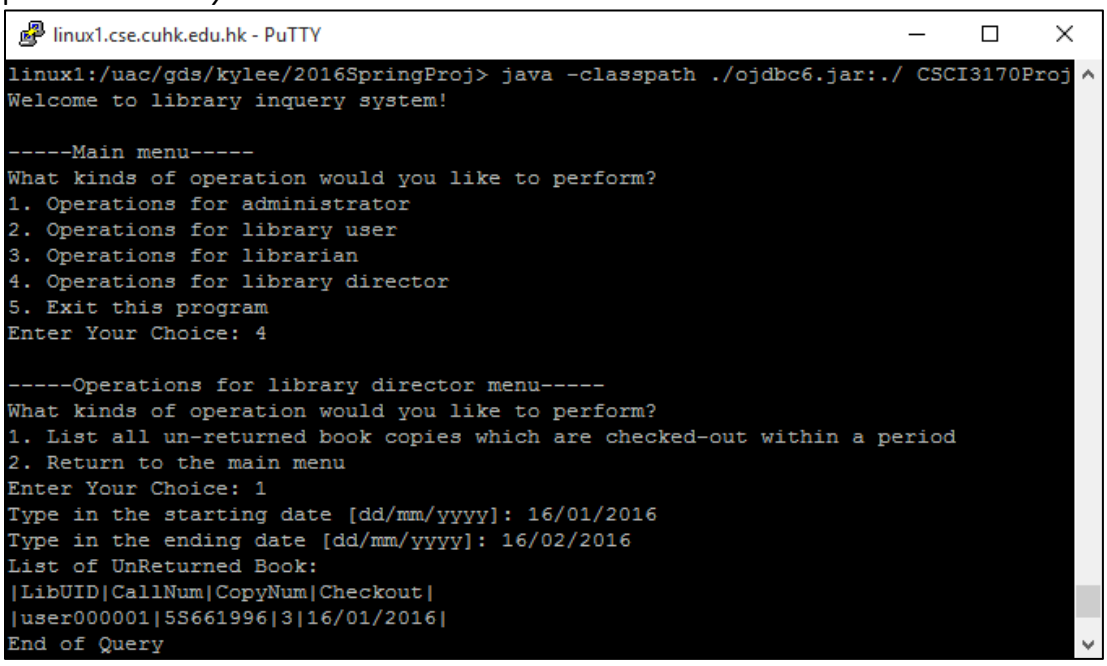

Figure 9: Expected input and output while listing out all unreturned books

5.5. Error Handling

If a run-time error occurs, the Library Inquiry System should output an information message in layman terms and in a new line as shown below.

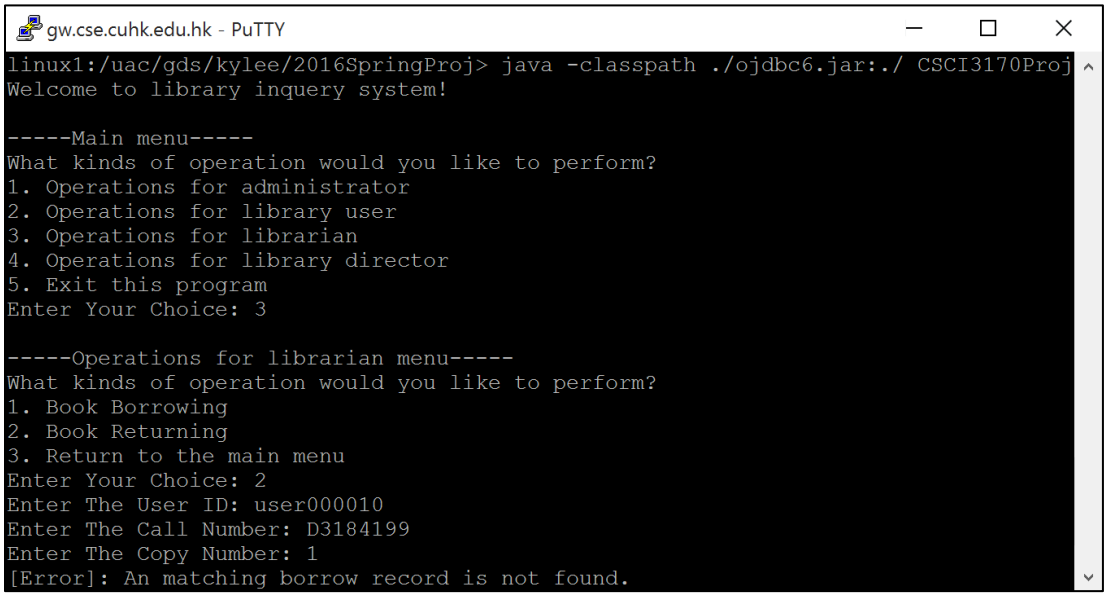

Figure 9: Expected input and output when an error occurs.

6. Grading Policy

The marks are distributed as follows:

Phase	Content	Mark Distribution
1	ER-diagram	10%
	Relational schema (based on your ER-diagram)	10%
2	Java application	80%

- There will be a mark deduction if your application is terminated unexpectedly during the demonstration.
- You are not allowed to modify any source code during the demonstration.
- All members in the same group will receive the same marks for the project. In order to encourage every student to participate in the project, a question about this project may be asked in the final examination.

7. Demonstration

- All groups need to sign up for a demonstration on their works for phase 2, the registration page would be posted on the course website later.
- All group members should attend the demonstration.
- The duration for the demonstration for each group is about 30 minutes.
- The Java application will be **complied** and **tested** in a Linux 64bit machine in the CSE department.
- The dataset used in the demonstration may be different from the dataset provided for testing.

8. Submission Methods

8.1. Phase 1

- Submit a PDF file (one copy for each group) to the collection box at eLearning platform.
- The PDF file should consist of your groups ER diagram, relational schema, the group number, the names and the student IDs of all group members of your group.

8.2. Phase 2

- Submit a ZIP file (one copy for each group) to the collection box at eLearning platform. The ZIP file should consist of all your source codes and a README file (README.txt), which contains:
  - The group number of your group
  - The name and the student ID of each group members of your group
  - List of files with description
  - Methods of compilation and execution