The Hong Kong Polytechnic University
Department of Electronic and Information Engineering

EIE3320 Lab 2: Library Admin System

(Deadline for Submission: Check the course information)

Important Note: This is a group project. Two students form a group. If you wish to do it on your own, one student per group is also fine but the scope will remain the same.

Expected Outcomes

- Understand the principles of object-oriented design.
- Apply Java in object oriented software development.
- Apply UML in object oriented software modelling.
- Apply object-oriented approach to developing computer software.
- Apply GUI and Data Structures with Java.
- Learn independently and be able to search for the information required in solving problems.
- Present ideas and finding effectively.
- Work in a team and collaborate effectively with others.

Mark distribution

1. Class Implementations: 20%
   - MyLinkedList class
   - MyQueue class
   - Book class
2. Basic Features: 60%
   - Basic features include all features described in the **Project Description**.
3. Additional features: 10%
   - Examples of additional features:
     - **Image Retrieval**: You may provide an image for each Book record and the image can be displayed in the page after clicking "More>>" button.
     - **Data Management**: You may save and retrieve the Book records to/from file or database.
     - Any other creative and significant features.
4. Report: 10%

Report requirement

Your report should contain (but not limited to) the following:

3.1 **Cover Page**: Course code/name, student IDs/names, "Lab 2 report" and date. Please type one person group if you work alone.

3.2 **Introduction**: A detailed description of the objectives and requirements of the program, and a brief description of the methodology.

3.3 **Methodology**: The methodology for implementing the program that includes

- How your team divides the work among the team members (for two person group)
- The schedule of implementing the program
- UML diagrams to show your object-oriented design:

    - UML Class Diagram and the specifications of the classes defined, and the public/private member functions/variables included;
    - The flow of execution using UML Activity Diagram.
- Describe the implementation and the use of the linked list and the queue classes. Include the EXTRACTED source code for explanation. DO NOT copy and paste the FULL source code to the report. Your source code files should be included in the submitted ZIP file.
- Describe the additional features implemented.

3.4 **Program Testing (through video demo)**

- Provide a link of video in the report to demonstrate the program testing. Make sure the link is accessible by other people. If the link is not accessible, marks will be deducted. (You may test whether the link is accessible by using another browser or incognito window.)
- Record a video to demonstrate the validations of your program and confirmed that it is running correctly. Show the execution results of your program.
- Show all the basic features and the additional features that you implemented.
- The video demo is used to assess the features implemented. Make sure all the features are demonstrated.
- You may just record the screen without verbal explanation.
- No need to provide screen captures in the report.

3.5 **Conclusion**

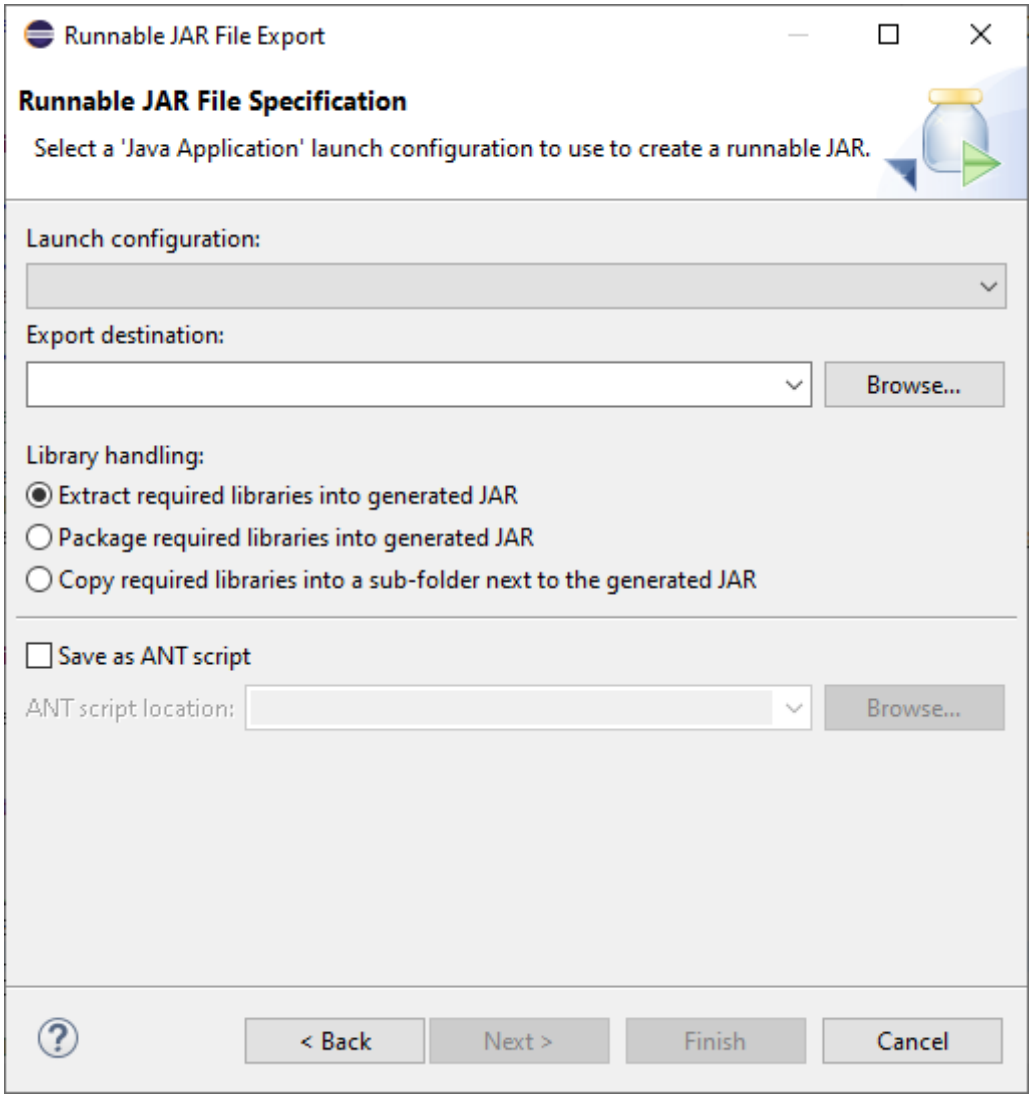
- Summarize the experience gained in the program

3.6 **Future Development**

- Indicate how your program can be extended.

Submission

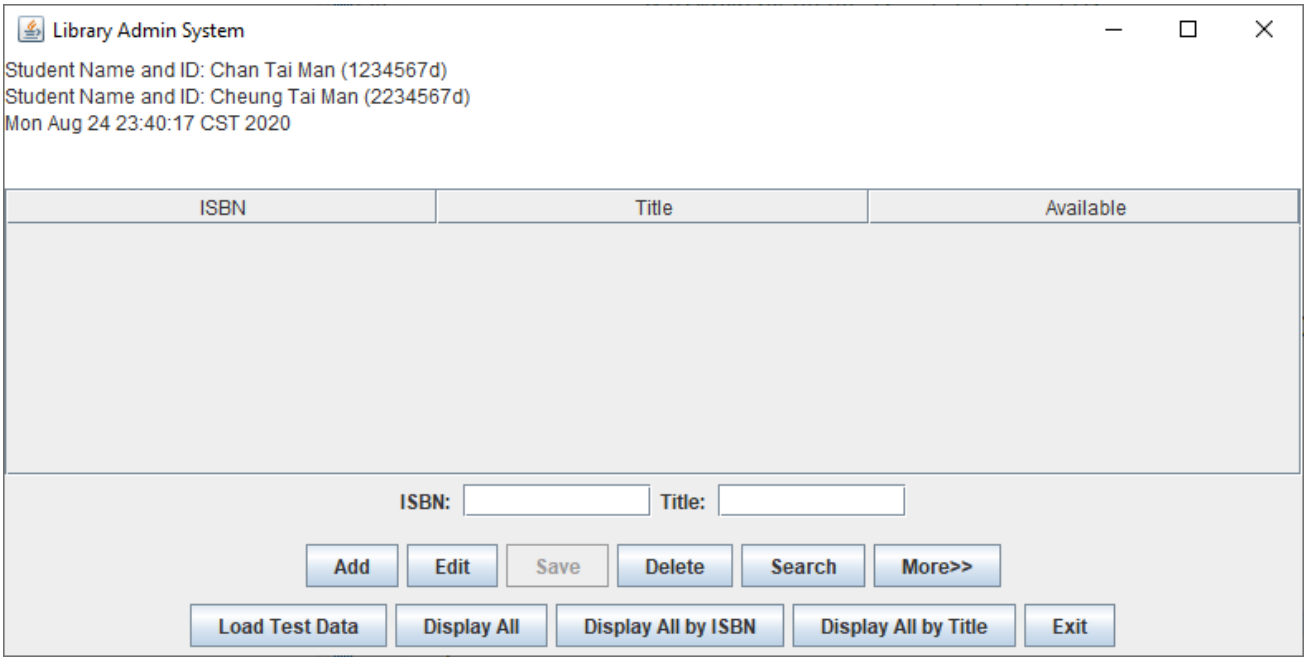
1. Submission includes the COMPLETE source code, the report with the video demo link.
2. Each team should compress your source code (all .java files), a runnable JAR file and report (PDF file) into ONE zip file and upload to Blackboard. The screen captures should contain your name(s), student number(s), and date and time at which you run your program.
3. Each team member should declare his/her responsibility in the report. Each member will be individually assessed based on the declared responsibility and the result obtained.
4. The report should be in PDF format. Please DO NOT include the complete source code in the report.
5. You should export your Project into a runnable JAR file. (Right click the project context root and select Export > Java > Runnable JAR file. Select the Main class (the class that contains the main() method to start the application) for Launch configuration. Browse to select the destination to save the export file. Select “Extract required libraries into generated JAR” for Library Handling.



6. It is compulsory to use a word processing tool to type your report. The font size must not be bigger than 12 or smaller than 10. Use 1.5 lines spacing on both sides of a page. Please convert your file into PDF format before submission.

Introduction

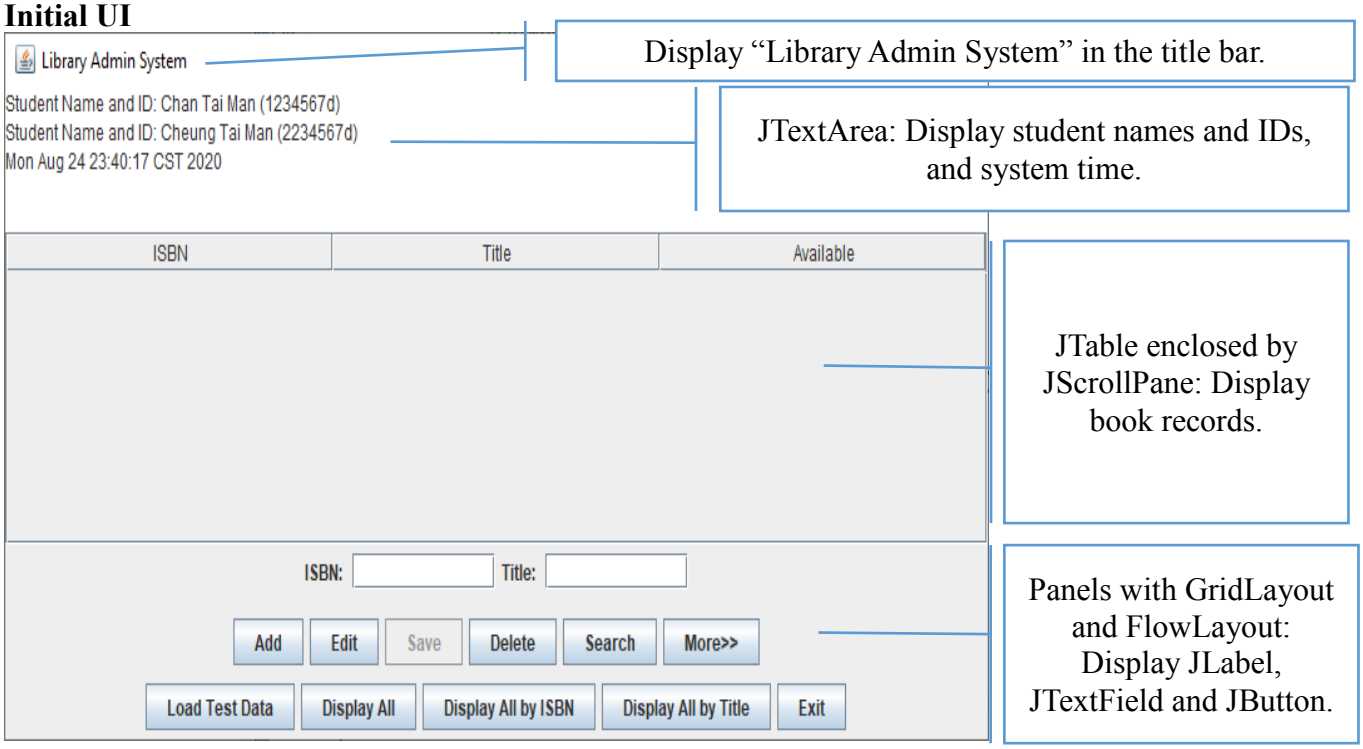
In this project, you are required to develop a Library Admin System by using Java. The system is a Java GUI application through which the library admin staff can (1) Add/Edit/Delete/Search/Display Book records and (2) Borrow/Return/Reserve books for the library users. Through this project, you will come across the design, implementation, and testing phases of a typical Java GUI application. The following shows the initial User Interface (UI) of the application.



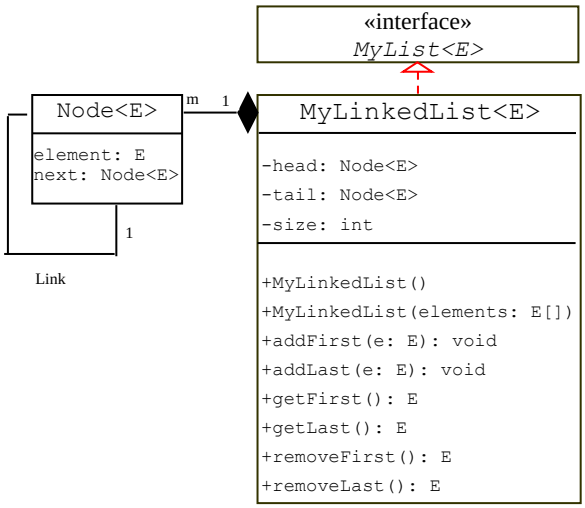
Project Description

In this project, you need to implement the following ***Basic Features***:

1. **Initial UI**

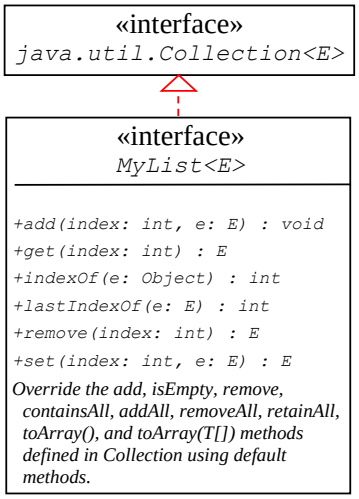


2. **MyLinkedList class**
   Instead of using the LinkedList class provided by java.util package, you should implement your own MyLinkedList class as described in Week10 lecture notes.



The head of the list.
The tail of the list.
The number of elements in the list.

Creates a default linked list.
Creates a linked list from an array of elements.
Adds an element to the head of the list.
Adds an element to the tail of the list.
Returns the first element in the list.
Returns the last element in the list.
Removes the first element from the list.
Removes the last element from the list.



Inserts a new element at the specified index in this list.
Returns the element from this list at the specified index.
Returns the index of the first matching element in this list.
Returns the index of the last matching element in this list.
Removes the element at the specified index and returns the removed element.
Sets the element at the specified index and returns the element being replaced.

Please download the templates for MyList and MyLinkedList from Blackboard. You may use the given TestMyLinkedList and TestMyLinkedListExtra to test your implementation of MyLinkedList class.

The sample output for TestMyLinkedList:

```
(1) [America]
(2) [Canada, America]
(3) [Canada, America, Russia]
(4) [Canada, America, Russia, France]
(5) [Canada, America, Germany, Russia, France]
(6) [Canada, America, Germany, Russia, France, Norway]
(7) [Poland, Canada, America, Germany, Russia, France, Norway]
(8) [Canada, America, Germany, Russia, France, Norway]
(9) [Canada, America, Russia, France, Norway]
(10) [Canada, America, Russia, France]
(11) CANADA AMERICA RUSSIA FRANCE
(12) iterator.next(): Canada
(13) iterator.next(): America
(14) iterator.next(): Russia
(15) list after iterator.remove(): [Canada, America, France]
Before clearing the list, the list size is 3
After clearing the list, the list size is 0
```

The sample output for TestMyLinkedListExtra:

```
[Tom, Susan, Kim, George, Peter, Jean, George, Jane, Denise, Jenny, Susan, Kathy, Jane]
Enter a name: Susan
Enter an index: 3
Susan is in the list? true
name at index 3 is George
Susan is at index 1
Susan is at last index 10
```

3. MyQueue class

Instead of using the LinkedList class or the PriorityQueue class provided by java.util package, you should implement your own MyQueue class as described in Week10 lecture notes.

MyQueue<E>	
-list: MyLinkedList<E>	
+enqueue(e: E): void	Adds an element to this queue.
+dequeue(): E	Removes an element from this queue.
+getSize(): int	Returns the number of elements from this queue.
+getList(): MyLinkedList<E>	Returns the list.

Please download the templates for MyQueue from Blackboard. You may use the given TestMyQueue to test your implementation of MyQueue class.

The sample output:

```
(1) Queue: [Tom]
(2) Queue: [Tom, Susan]
(3) Queue: [Tom, Susan, Kim, Michael]
(4) Tom
(5) Susan
(6) Queue: [Kim, Michael]
   The size of the queue is 2
```

4. Book class

Implement a Book class with the following private fields. Create getter and setter methods for each field.

```
private String title;      // store the title of the book
private String ISBN;   // store the ISBN of the book
private boolean available;   // keep the status of whether the book is available;
                             // initially should be true
private MyQueue<String> reservedQueue;    // store the queue of waiting list
```

The system should maintain a linked list of Book objects using MyLinkedList<Book>.

Please download TestBook class from Blackboard to test your implementation of Book class.

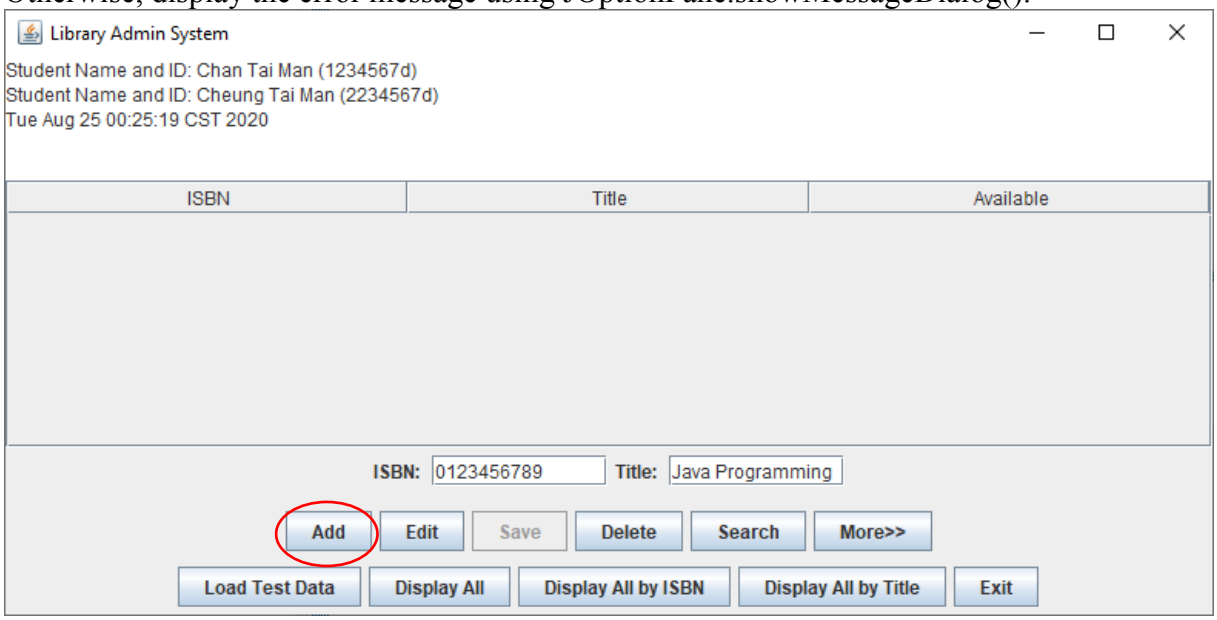
The sample output:

```
11111111 Book1: available
22222222 Book2: available
33333333 Book3: Not available
    Reserved queue: Borrower1 Borrower2 Borrower3
44444444 Book4: available
```

5. Add Book

If the ISBN and Title text fields are not empty, and the ISBN does not exist in the current linked list of Book objects, then create the Book object and add to the linked list. Display the newly created Book in the JTable and clear the textfields.

Otherwise, display the error message using JOptionPane.showMessageDialog().



6. Load Test Data

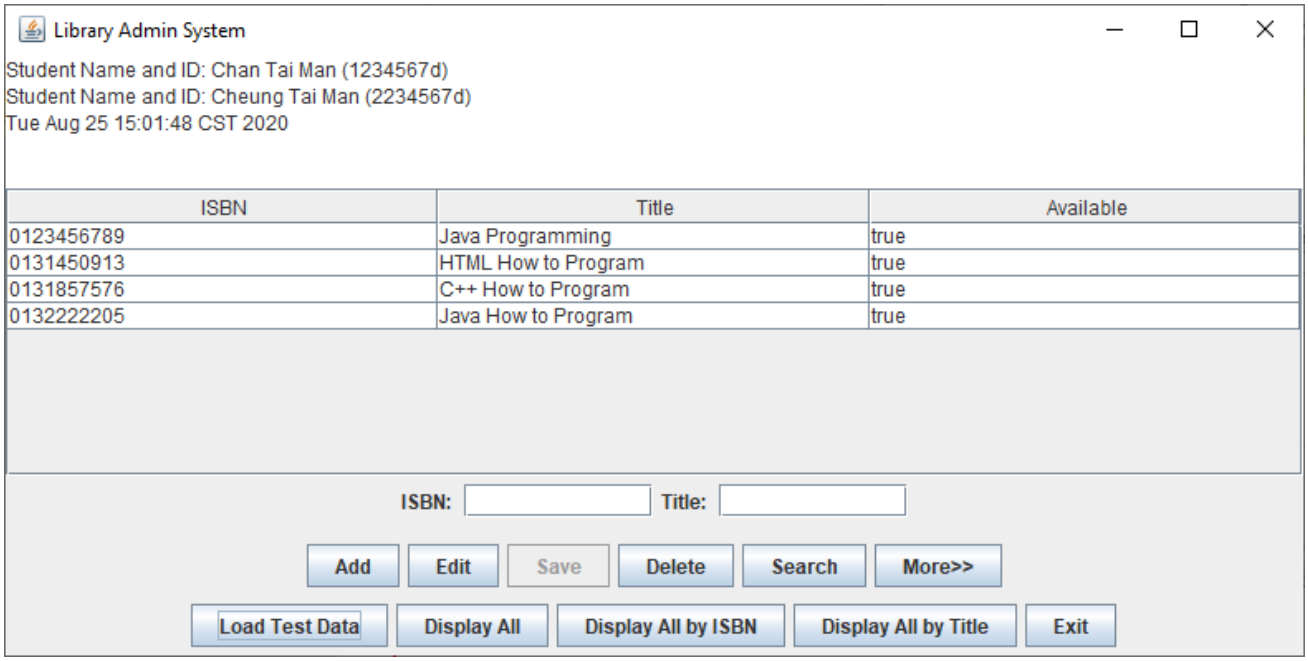
On clicking "Load Test Data" button, the following 3 books will be appended to the current linked list.

Title: "HTML How to Program"
ISBN: "0131450913"

Title: "C++ How to Program");
ISBN: "0131857576");

Title: "Java How to Program");
ISBN: "0132222205");

Then, all book records will be displayed in the JTable. The textfields are cleared.



- Clicking "Load Test Data" button again would display the error message since the books are already in the database.
- Delete "HTML How to Program" and click "Load Test Data" button again. The "HTML How to Program" book will be added back and displayed in the table while the other two books cannot be added with error message displayed.

7. Edit Book and Save

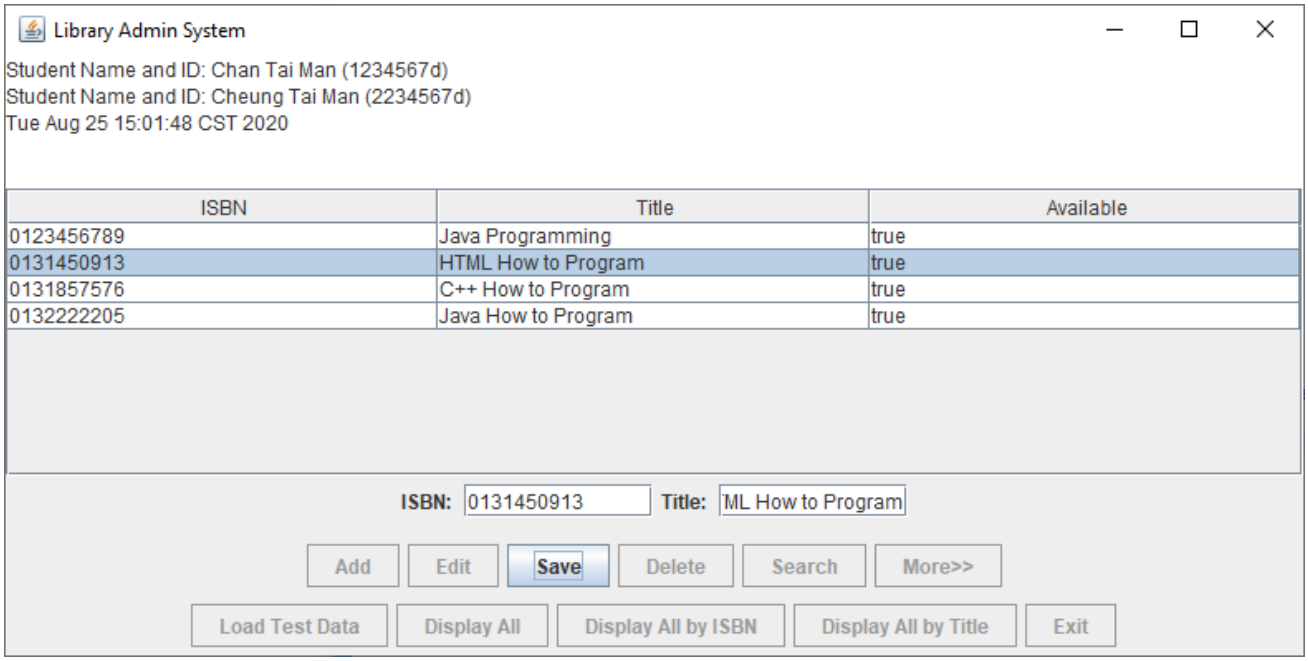
There are two ways to select book for edit:

(1) Click the book record in the JTable, then the Book ISBN and Title will be displayed in the textfields. Click the "Edit" button.

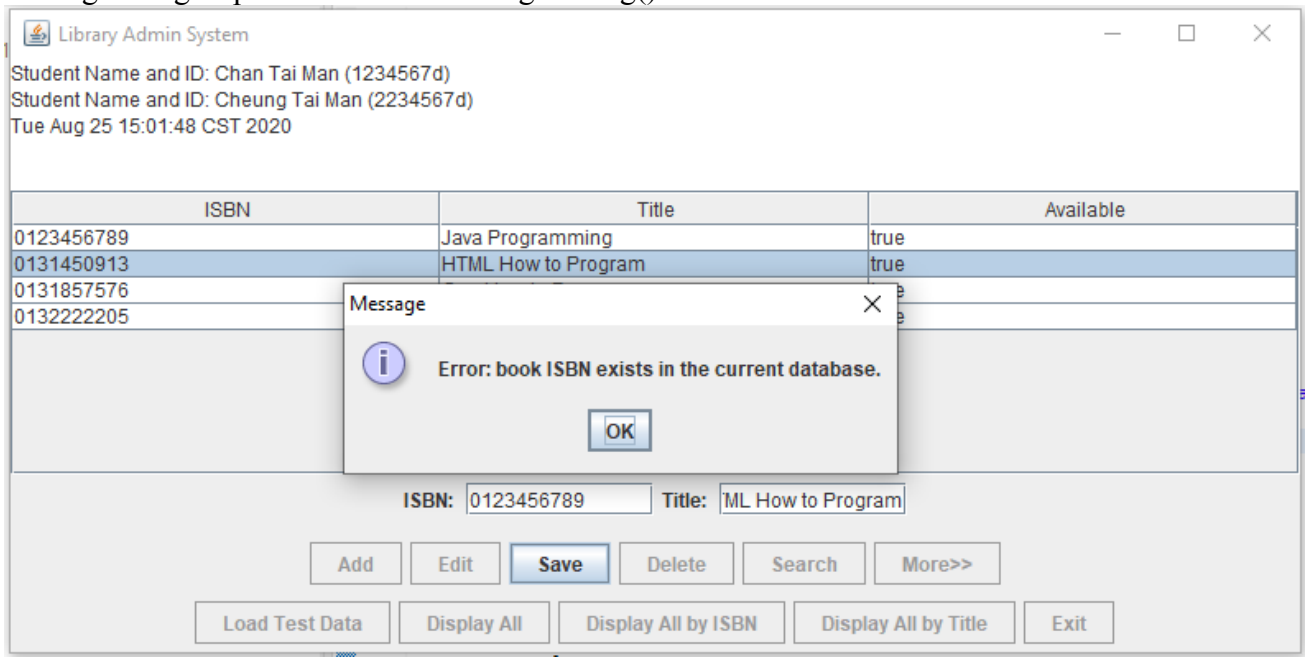
(2) Input the ISBN of the book and click "Edit" button, the title of the book will be displayed in the textfield.

If the linked list is empty or the linked list does not contain the input ISBN, display the error message using JOptionPane.showMessageDialog().

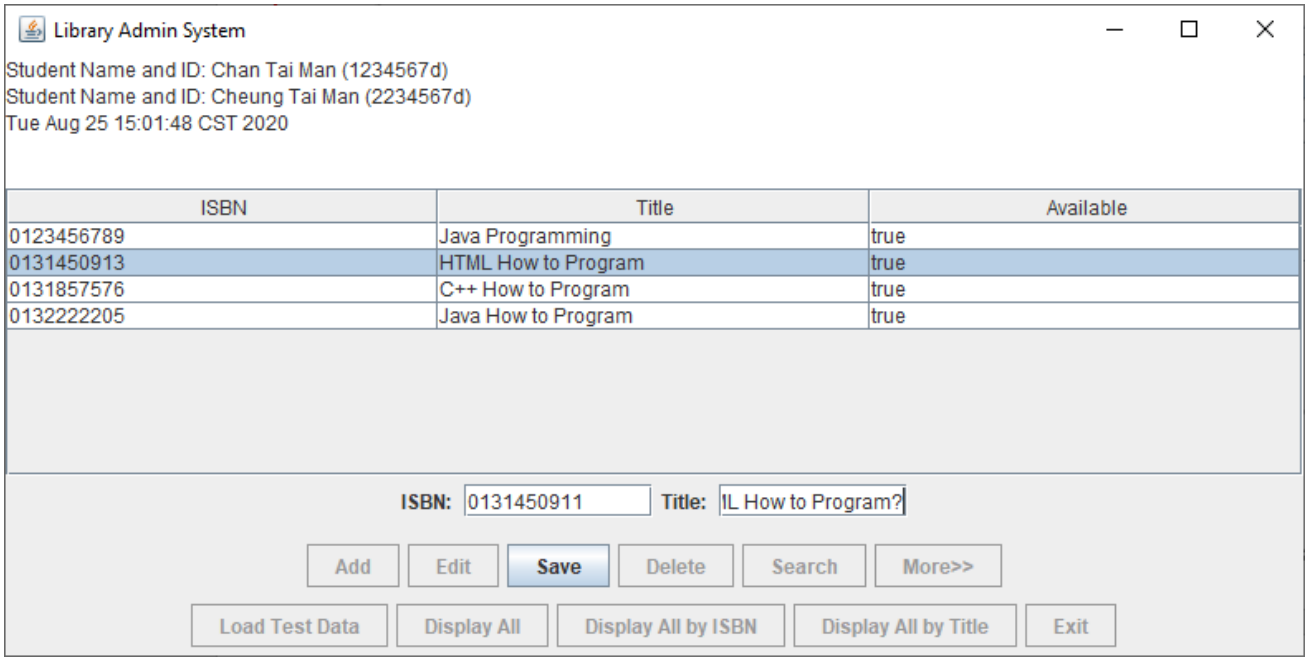
Otherwise, the "Save" button becomes enabled while all the other buttons are disabled.

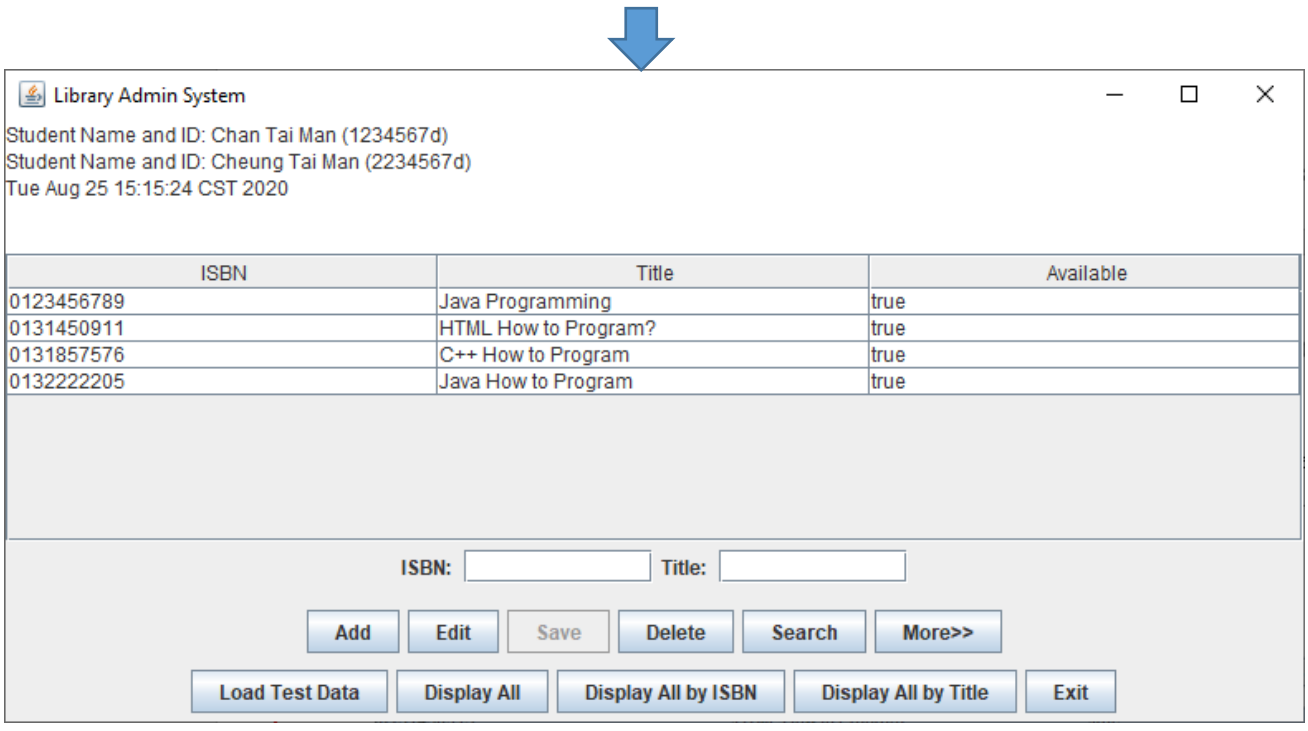


User may now edit the ISBN or/and the book title, and then click the "Save" button.
If the ISBN is changed and the modified ISBN exists in the current linked list, display the error message using JOptionPane.showMessageDialog().



Otherwise, save the modified information to the Book object.
Afterwards, display all records with the updated information. Disable the "Save" button and enable all the other buttons. Clear the text fields.





8. Delete Book

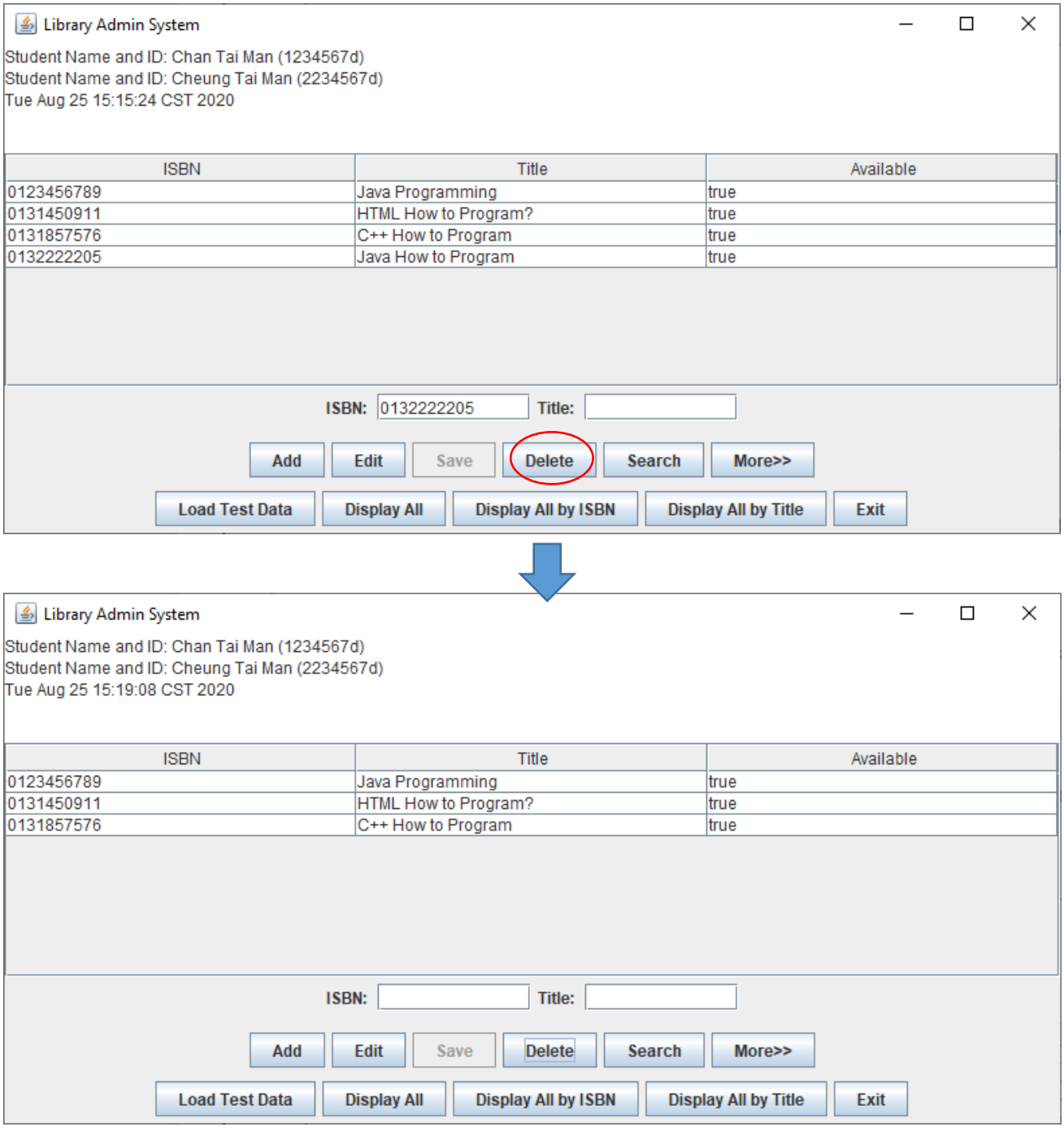
There are two ways to select book for delete:

(1) Click the book record in the JTable, then the Book ISBN and Title will be displayed in the text fields. Click the "Delete" button.

(2) Input the ISBN of the book and click "Delete" button.

If the linked list is empty or the linked list does not contain the input ISBN, display the error message using JOptionPane.showMessageDialog().

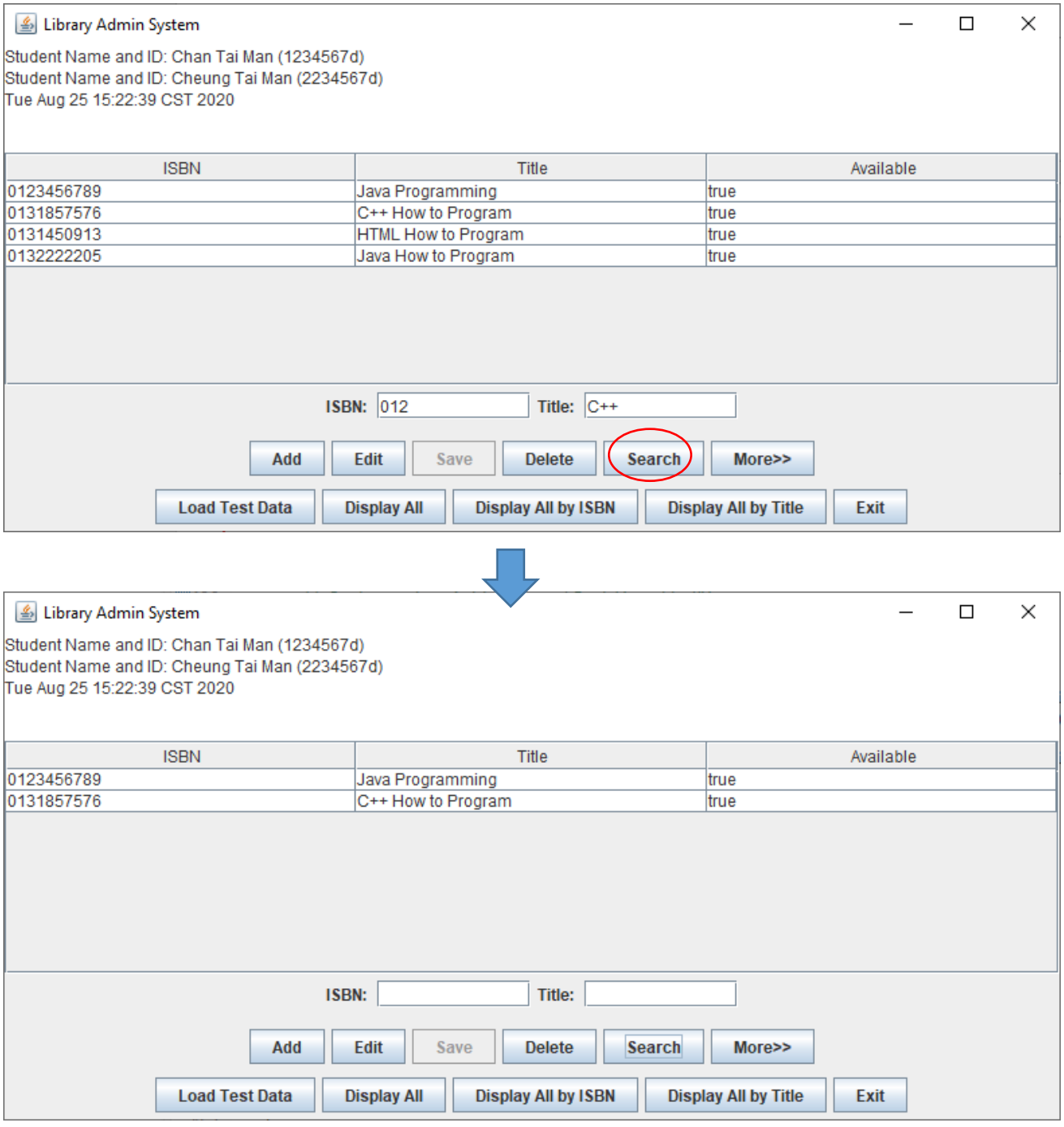
Otherwise, remove the selected book from the linked list.
Display all records with the selected book removed and clear the text fields.



9. Search Book

Input the keywords to be searched on ISBN text field and/or Title text field.

On clicking the "Search" button, if the text fields are not empty, display the Book record that contains the keyword in the ISBN text field or the Title text field.

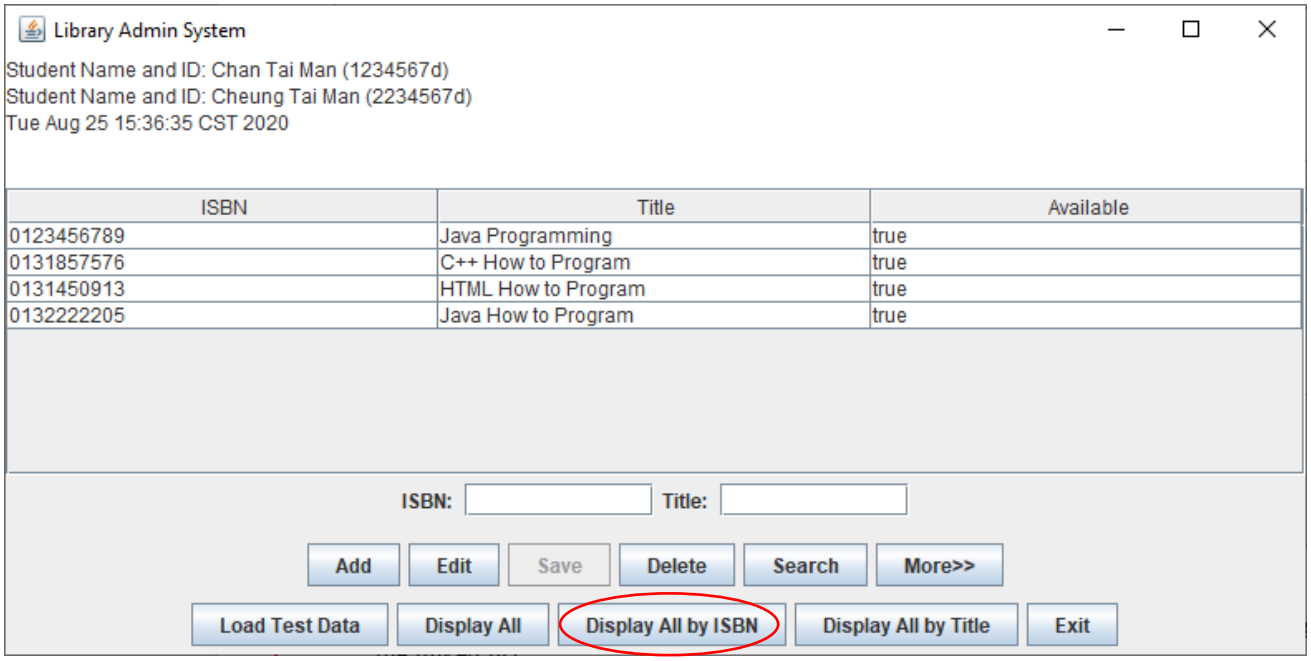


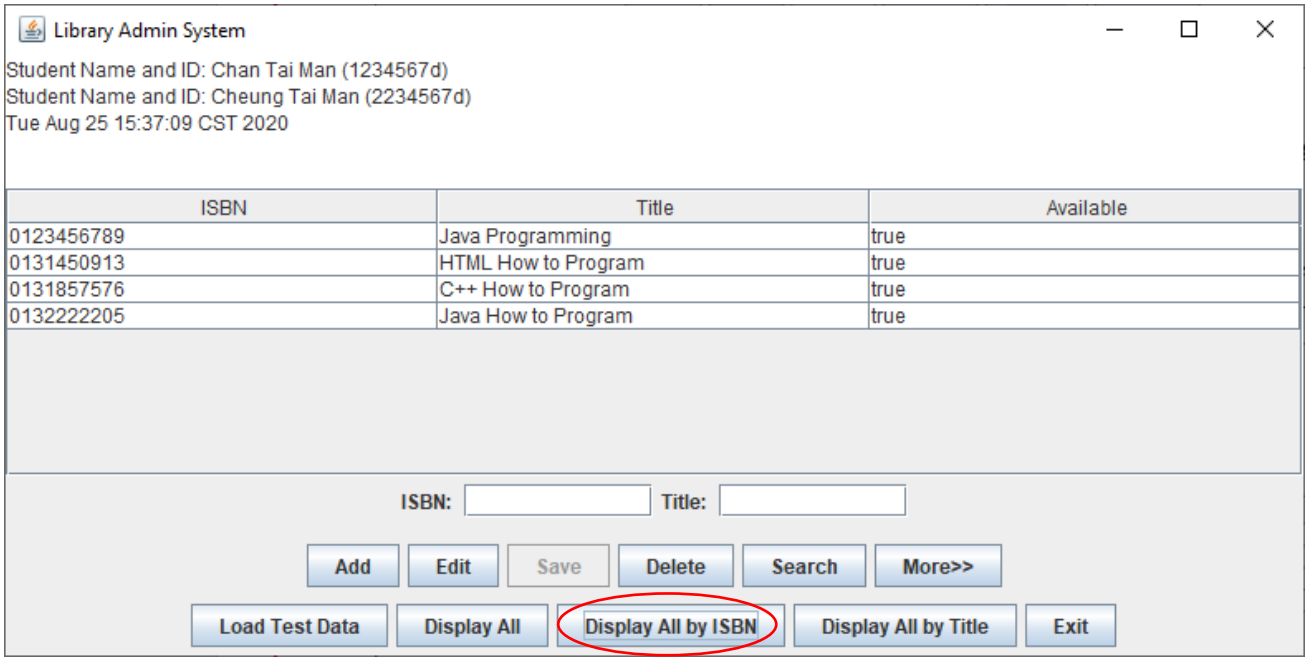
10. Display All Books

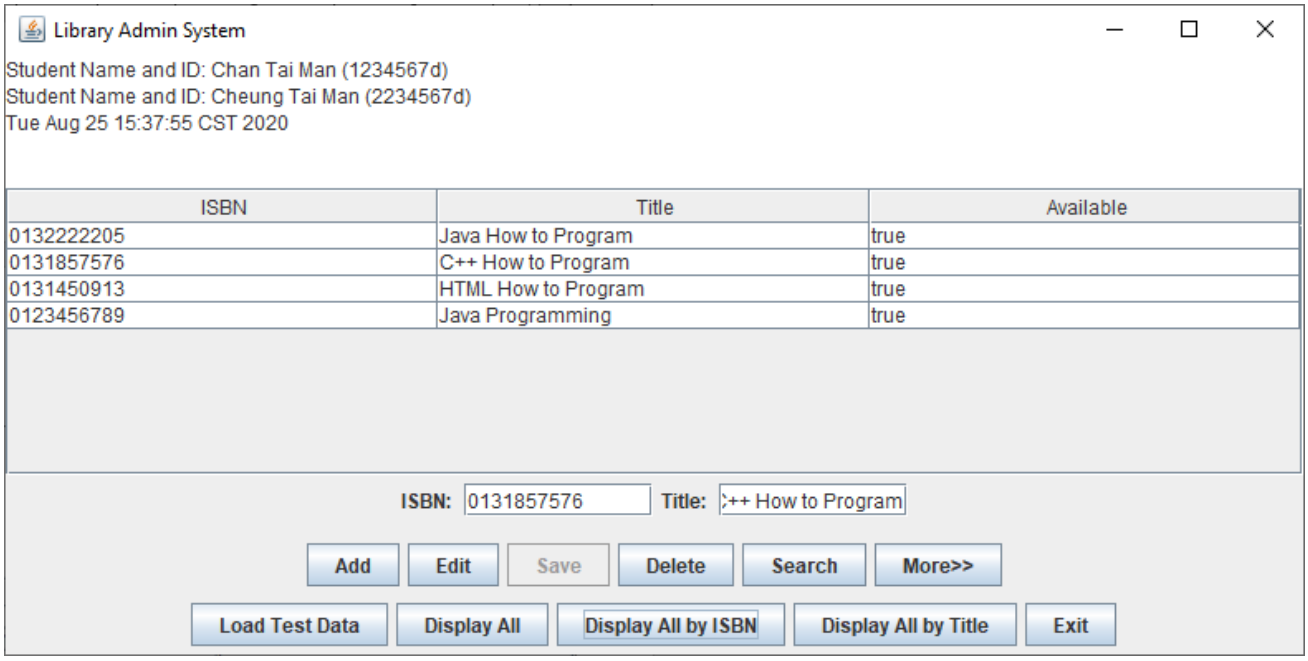
On clicking "Display All" button, display all book records following the order of adding nodes to the linked list.

11. Display All Books by ISBN

On clicking "Display All by ISBN" button, display all book records in ascending order of Book ISBN. If the button is clicked again, display all book records in reversed order on subsequent clicks of the button.


Library Admin System
Student Name and ID: Chan Tai Man (1234567d)
Student Name and ID: Cheung Tai Man (2234567d)
Tue Aug 25 15:36:35 CST 2020
ISBN
Title
Available
0123456789
Java Programming
true
0131857576
C++ How to Program
true
0131450913
HTML How to Program
true
0132222205
Java How to Program
true
ISBN:
Title:
Add
Edit
Save
Delete
Search
More>>
Load Test Data
Display All
Display All by ISBN
Display All by Title
Exit


Library Admin System
Student Name and ID: Chan Tai Man (1234567d)
Student Name and ID: Cheung Tai Man (2234567d)
Tue Aug 25 15:37:09 CST 2020
ISBN
Title
Available
0123456789
Java Programming
true
0131450913
HTML How to Program
true
0131857576
C++ How to Program
true
0132222205
Java How to Program
true
ISBN:
Title:
Add
Edit
Save
Delete
Search
More>>
Load Test Data
Display All
Display All by ISBN
Display All by Title
Exit



12. Display All Books by Title

On clicking "Display All by Title" button, display all book records in ascending order of Book Title. If the button is clicked again, display all book records in reversed order on subsequent clicks of the button.

13. Select Book with more options

13.1. Display a new page with more options

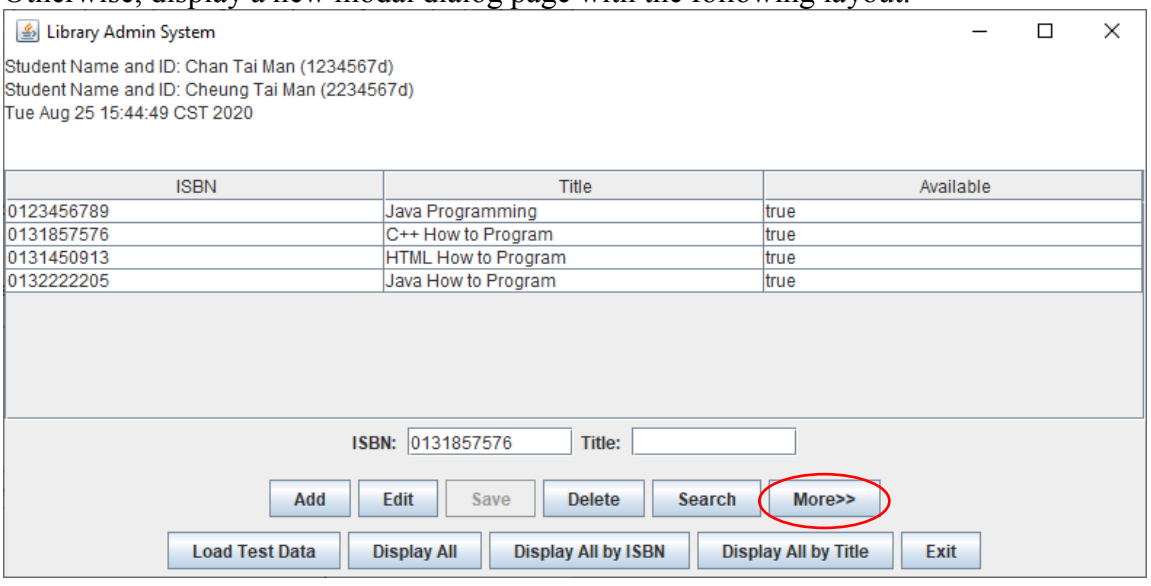
There are two ways to select book with more options:

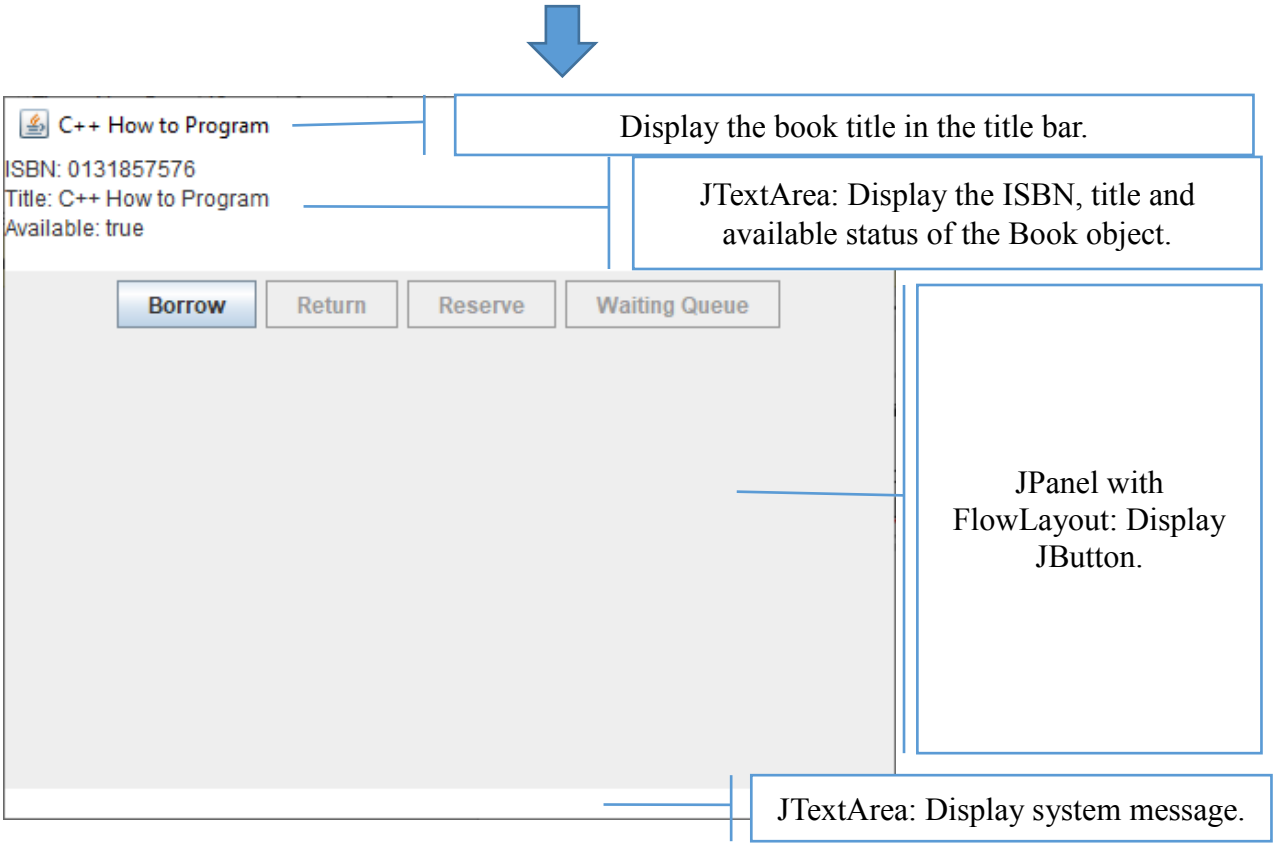
(1) Click the book record in the JTable, then the Book ISBN and Title will be displayed in the textfields. Click the "More" button.

(2) Input the ISBN of the book and click "More" button.

If the ISBN text field is empty, display the error message using JOptionPane.showMessageDialog().

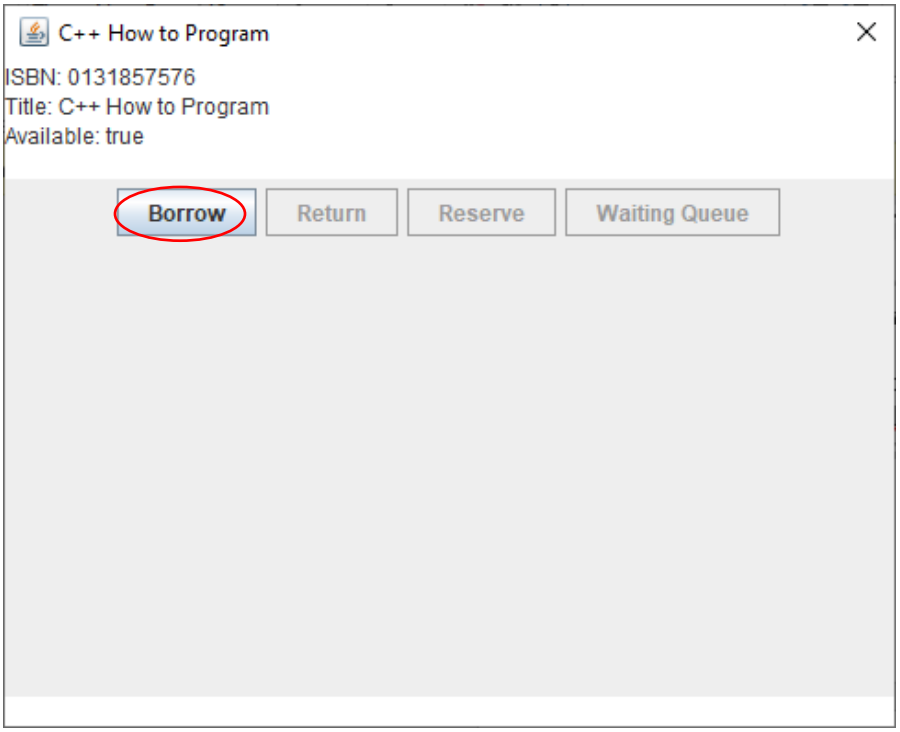
Otherwise, display a new modal dialog page with the following layout.

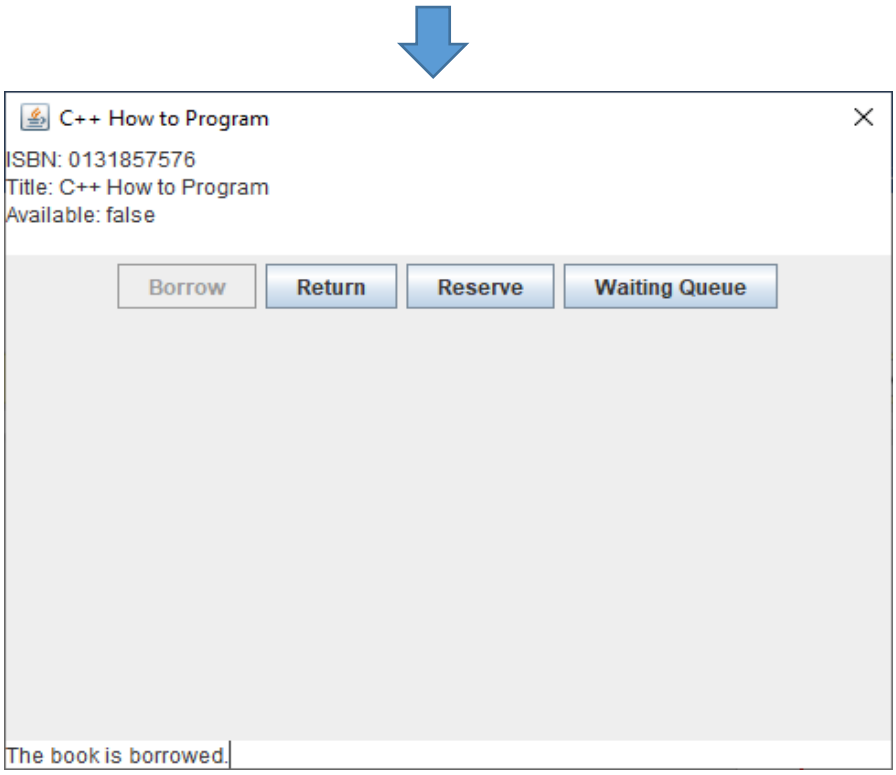




13.2. Borrow Book

If the Book object is available, the “Borrow” button is enabled while all the other buttons are disabled.
On clicking the “Borrow” button, the book available status is set to false.
The “Borrow” button becomes disabled while all the other buttons becomes enabled.
Display the updated available status at the top and display the message "The book is borrowed." at the bottom.

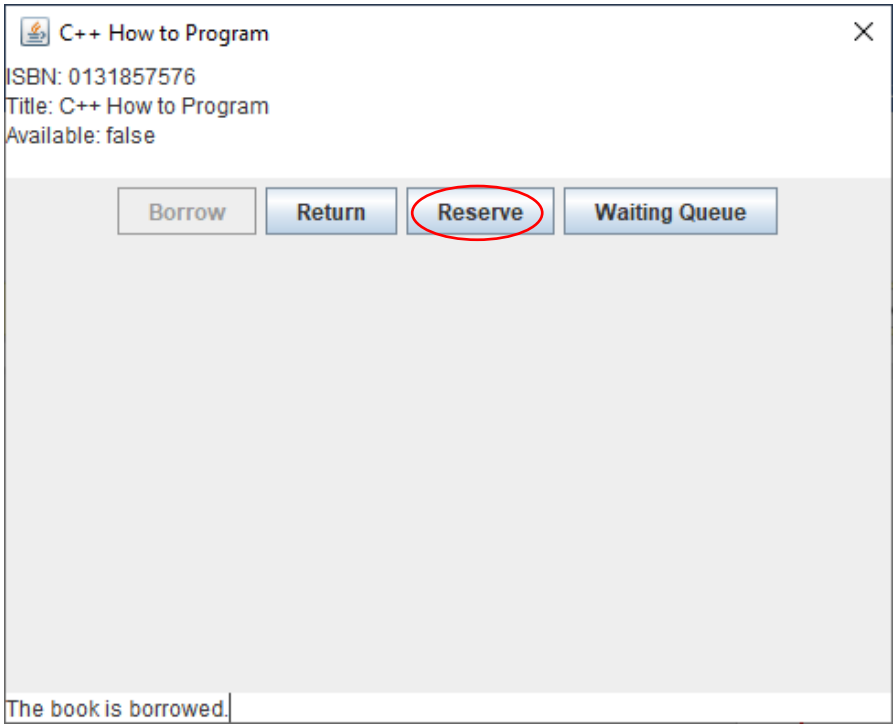


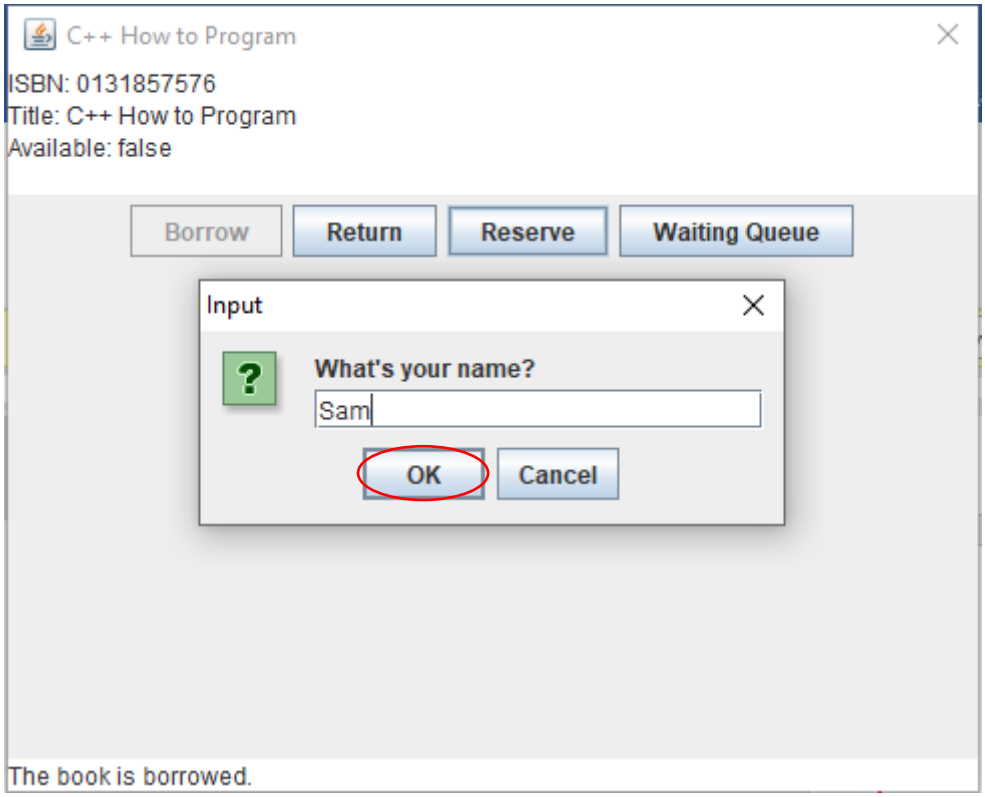


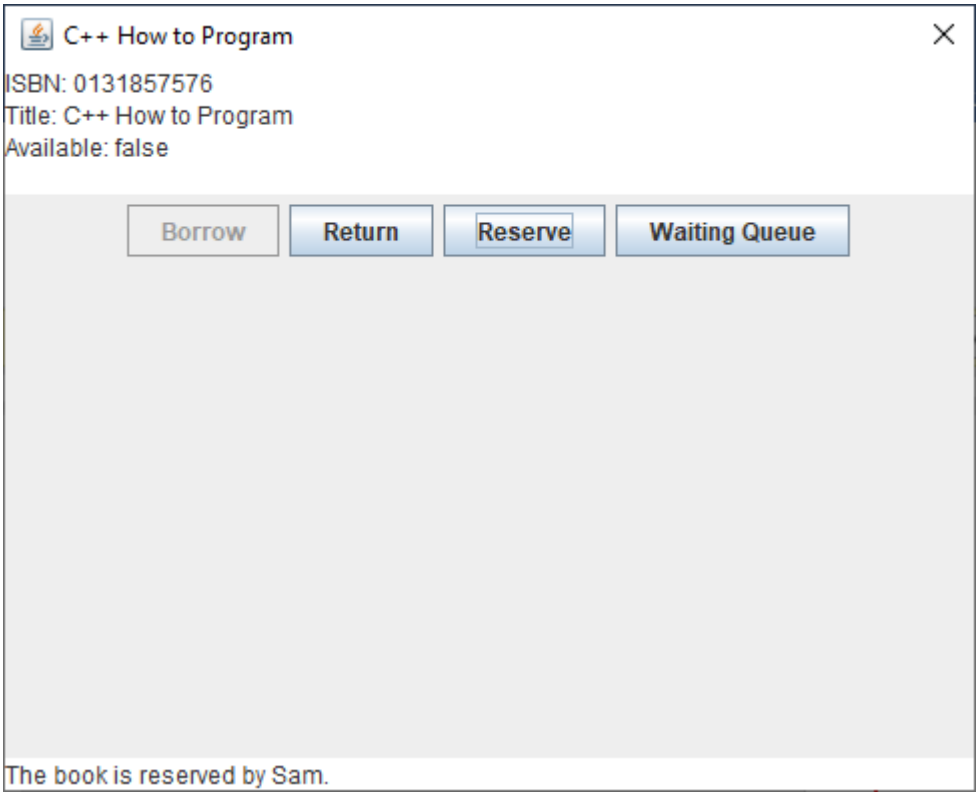
13.3. Reserve Book

If the book is not available, the library admin staff cannot borrow the book for the library user. Instead, he/she can return the book or reserve the book for the library user.

Suppose now a user wants to reserve the book, the library admin staff should click the "Reserve" button. An input dialog using JOptionPane.showInputDialog() would be popped up to prompt for the user's name. The library admin staff should input the user's name, which will then be added to the reservedQueue of the Book object. The message "The book is reserved by *the user's name*." will be displayed at the bottom.


C++ How to Program
ISBN: 0131857576
Title: C++ How to Program
Available: false
Borrow
Return
Reserve
Waiting Queue
The book is borrowed.

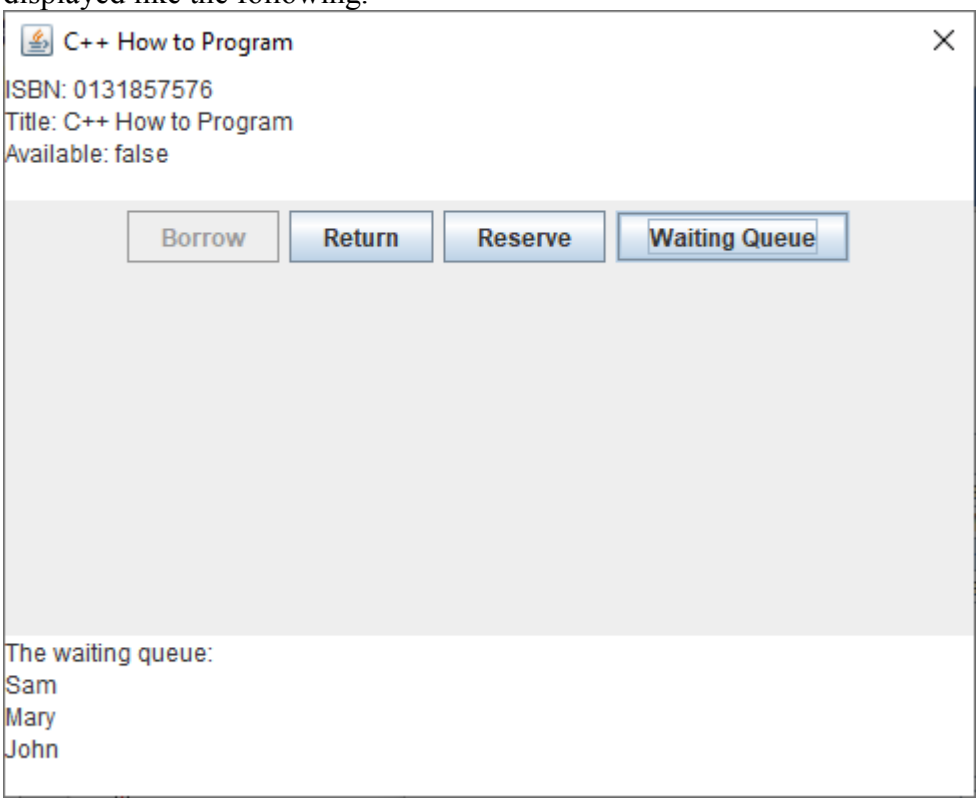

C++ How to Program
ISBN: 0131857576
Title: C++ How to Program
Available: false
Borrow
Return
Reserve
Waiting Queue
Input
What's your name?
Sam
OK
Cancel
The book is borrowed.



13.4. Display waiting queue

On clicking "Waiting Queue" button, the reserved queue of the Book object will be displayed at the bottom.

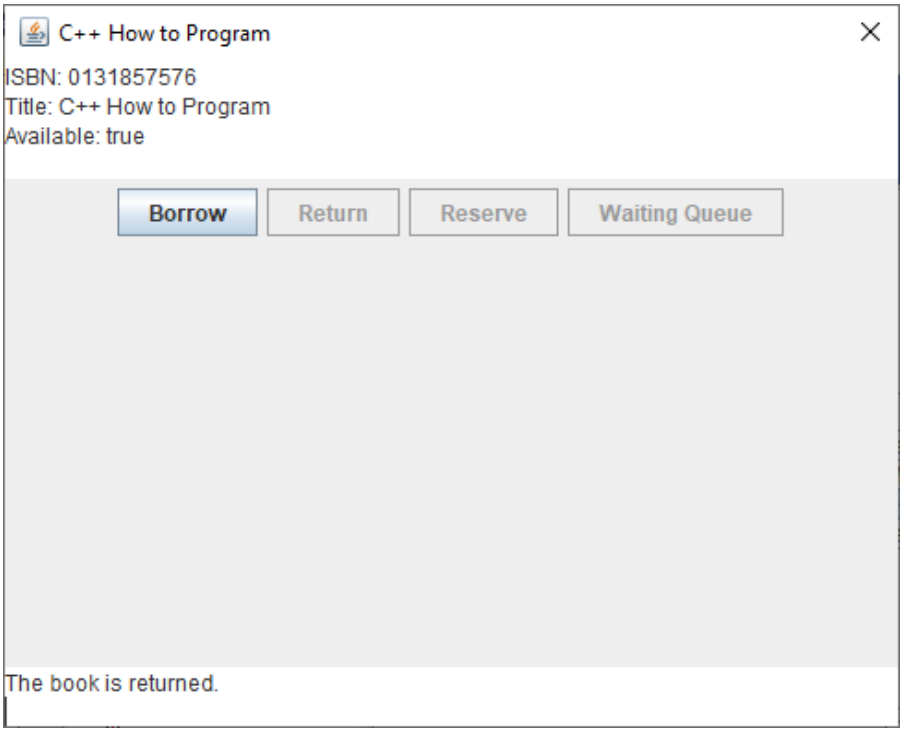
Suppose Sam has reserved the book, followed by Mary and John, the message will be displayed like the following.



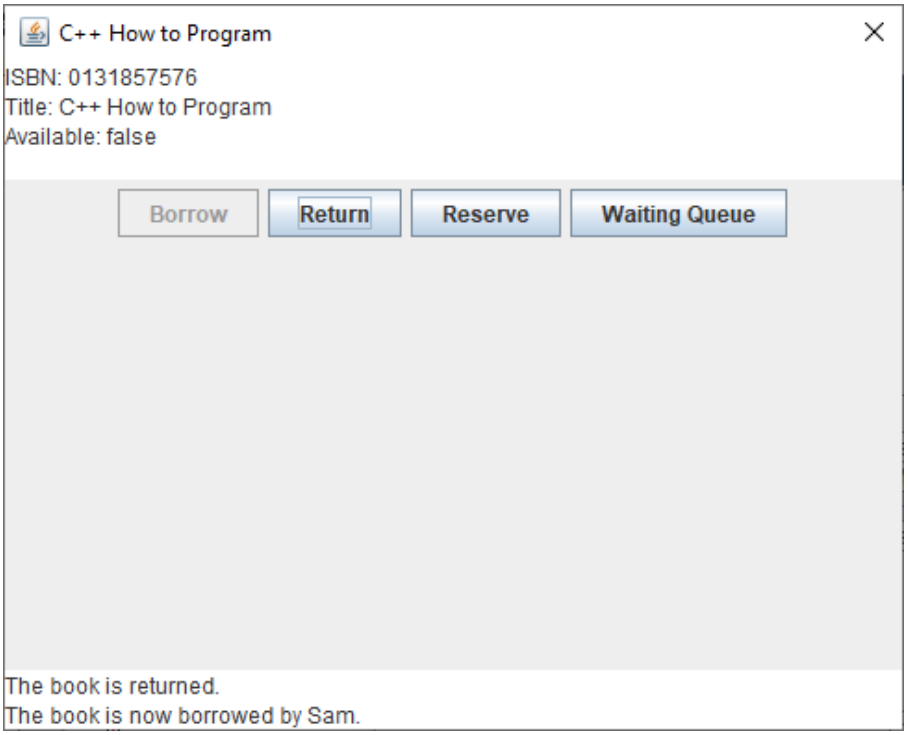
13.5. Return Book

On clicking the “Return” button, the message "The book is returned." will be displayed at the bottom.

If there is no user in the reserved queue of the Book object, the available status will be turned to true. The “Borrow” button becomes enabled and all the other buttons become disabled.



Otherwise, the available status remains to be false. The book will then be borrowed by the head user in the reserved queue, who is then removed from the queue. The message "The book is now borrowed by *the user’s name*." will also be displayed.



14. Exit application

On clicking the “Exit” button or the “X” button on the top right corner of the window, the application will be closed.

15. Additional Features

You may decide on which additional features to be implemented.

References

- How to Use Tables

https://docs.oracle.com/javase/tutorial/uiswing/components/table.html

- Class JTable

https://docs.oracle.com/javase/7/docs/api/javax/swing/JTable.html

- How to Make Dialogs

https://docs.oracle.com/javase/tutorial/uiswing/components/dialog.html

- Class JDialog

https://docs.oracle.com/javase/7/docs/api/javax/swing/JDialog.html