# Xi’an JIAOTONG-LIVERPOOL UNIVERSITY

西 交 利 物 浦 大 学

# Coursework Submission Cover Sheet

|  |  |  |
| --- | --- | --- |
| Name | Jin(Surname) | Minhao(Other Names) |
| Student Number | 1717576 | |
| Programme | Information and Computing Science | |
| Module Title | Data Structures | |
| Module Code | CSE104 | |
| Assignment Title | Assignment 2 | |
| Submission Deadline | 2019-05-24 12:00pm | |
| Module Leader | Prof. Steven Guan | |

By uploading or submitting this coursework submission cover sheet, I certify the following:

* I have read and understood the definitions of PLAGIARISM, COLLUSION, and the FABRICATION Of DATA, as outlined in the Undergraduate Student Handbook of Xi’an Jiaotong-Liverpool University and as posted on the University Website.
* This work is my own, original work produced specifically for this assignment. It does not misrepresent the work of another person or institution as my own. Additionally, it is a submission that has not been previously published, or submitted to another module.
* This work is not the product of unauthorized collaboration between myself and others.
* This work is free of embellished or fabricated data.

I understand that PLAGIARISM, COLLUSION and the FABRICATION OF DATA are serious disciplinary offences. By uploading or submitting this cover sheet, I acknowledge that I am subject to disciplinary action if I am found to have committed such acts.

Signature ……*.*…………………………………… Date ……*………………………………*……………

|  |  |  |  |
| --- | --- | --- | --- |
| For Academic Office use: | Date Received | Days Late | Penalty |
|  |  |  |

## Feedback on the strength of the work

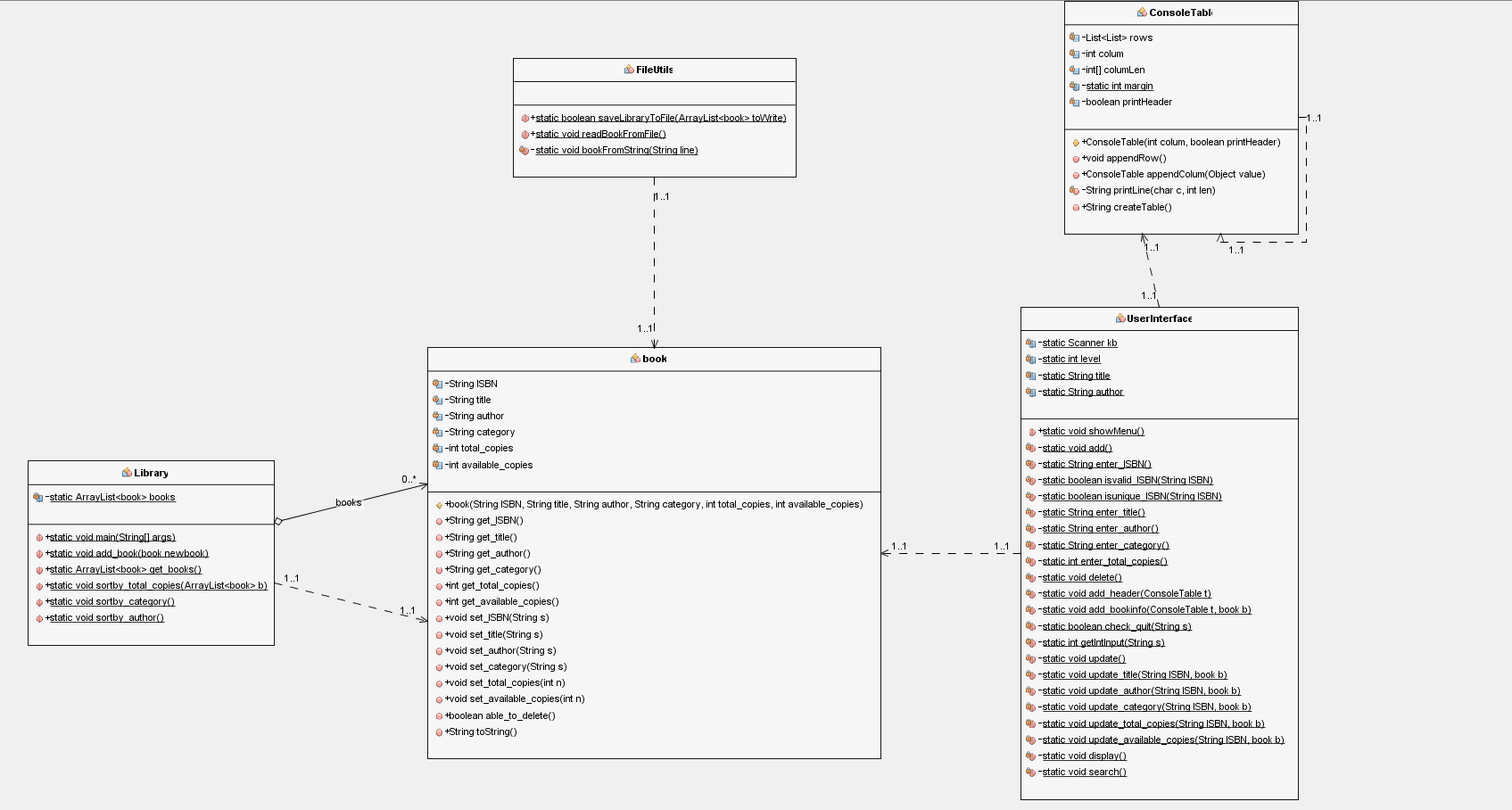
Feedback on the weakness that needs to be improved

**1st Marker Date Mark 2nd Marker Date Mark**

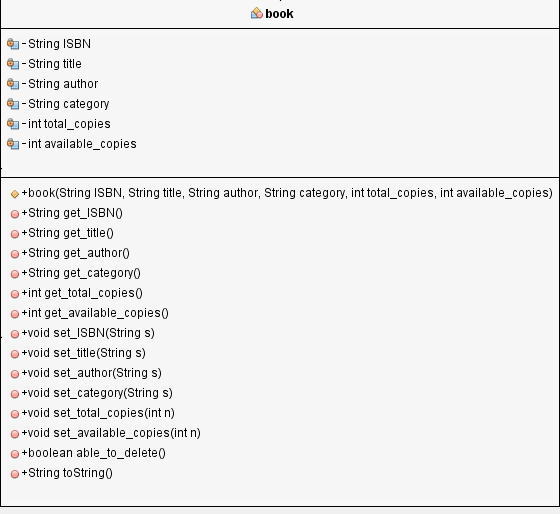
**(if applicable)**

***Students:*** *Please start your assignment on the next page.*

UML:



Data structure of the ‘book’ object



Test cases

1. Check whether the input is “quit”

Time complexity: O(1)

Pseudo code:

boolean check\_quit(String s)

Set s to its lowercase

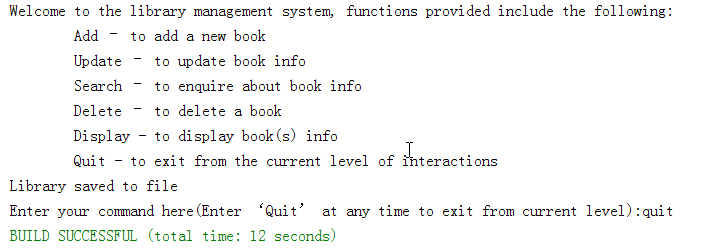
if(s==”quit” and level==1) return true

else if(s==”quit” and level==0) stop the whole program

else return false

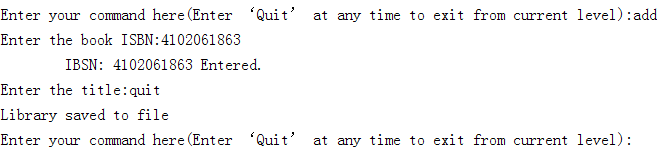
Explanation: I divide library management system into two levels, the main menu is at the level 0 and all the other functions are all settled on level 1. If the user input the “quit” at level 0, the whole program ends. In the other case, if the user input “quit” at level 1, then the program will go back to level 0 and show the main menu.

Test case 1: input “quit” at level 0.



The program ends.

Test case 2: input “quit” at level 1.



The program does not end but go back to the main menu, waiting for the next command.

2. Check whether the input ISBN is valid and unique

isvalid\_ISBN time complexity: O(1)

isunique\_ISBN time complexity: O(n)

(n is the number of books already in the library system)

Pseudo code:

boolean isunique\_ISBN(String ISBN)

for (book b : books)

begin

if (b’s ISBN==ISBN)

return false;

end

return true

String enter\_title()

while (true)

begin

print("Enter the title:")

User inputs the title

if (check\_quit(title))

return "quit";

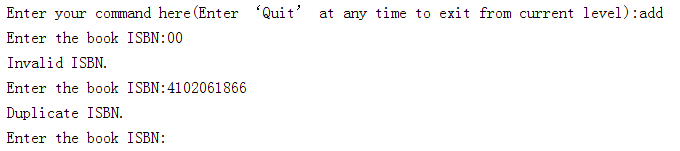
println("\tTitle: " + \_title + " Entered.");

return \_title;

end

Explanation: If the ISBN is invalid or it is not unique in the library system, they are not allowed to be input.

Test case: supposing a book with ISBN 4102061866 is already in the library system.



The program will inform the user that the ISBN entered is invalid or duplicate, then it will let the user continue to enter ISBN until it is acceptable.

3. Check whether the input integer is valid

Pseudo code:

private static int getIntInput(String s) {

int input = 0;

while (true)

begin

if (check\_quit(s))

break;

try

convert s to an Integer input

if (input < 0)

println("That is not a positive number please try again.");

input = getIntInput(User’s input);

catch (NumberFormatException e)

println("That is not a number please try again.");

input = getIntInput(kb.nextLine());

finally

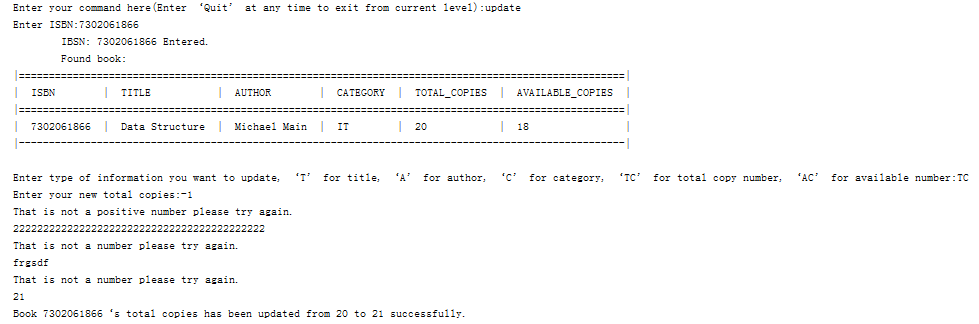
return input;

end

return input;

Explanation: At first, try casting the String to an integer. If this fails or the integer is negative, user will be told to enter again until a valid integer is entered.

Test case: in the update function, the total copies of one book will be changed.



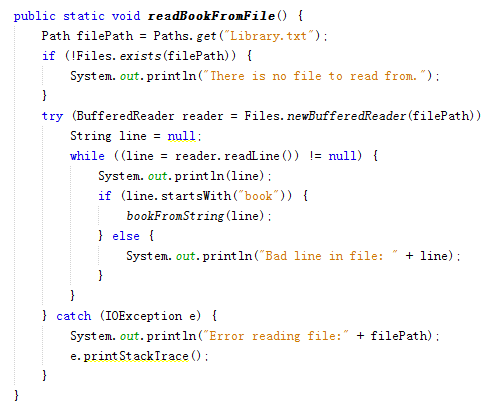
As you can see in the figure above, all the negative and invalid integer are all unacceptable. The program shows that regardless of the invalid input, it successfully changes the total copies from the original one to 21.

Read and write from a .txt file

The fileutils class is mainly responsible for writing and reading all the data from a .txt file which act as a database of the library system.

Read from the .txt file

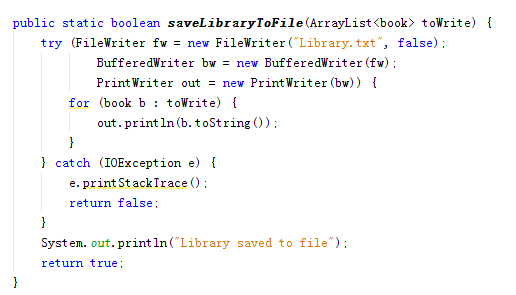
Time complexity: O(n)



Explanation: As the program runs, it will load all the data from that .txt at first through this method. Each line records all the information of one book. The format for recording all the books are the same so that it will be easy to extract all the key attributes of the book. After that, a ‘book’ object will be created with these attributes and they are saved in an arraylist as the original data source for the library system.

Write into the .txt file:

Time complexity: O(n)



Explanation: After adding a new book or updating a book, the data should be recorded. Therefore, after one function in the main menu is finished, all the data saved in the book arraylist will be written into the .txt file, so that the database will be refreshed in time.

One sample of the database:

