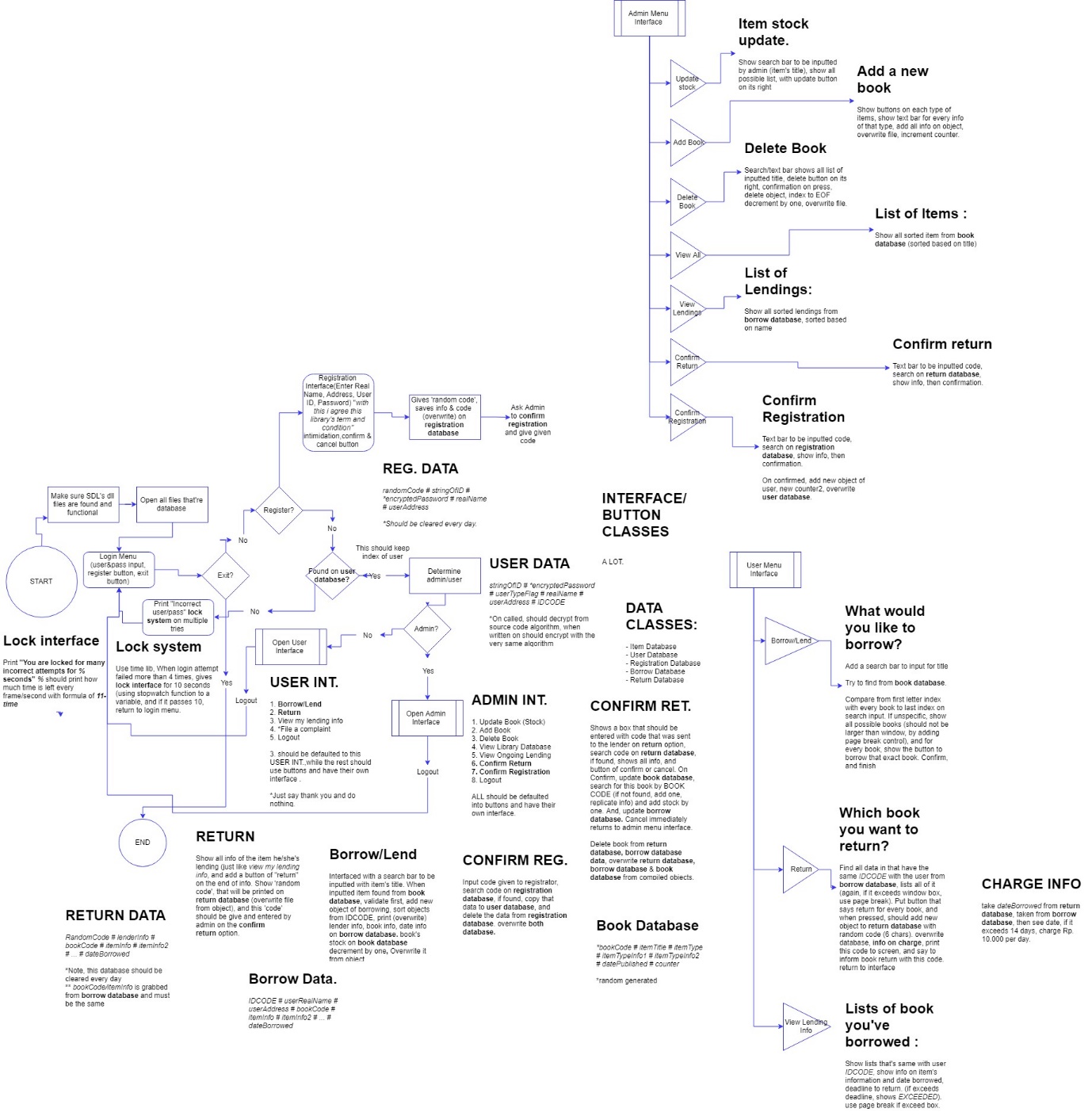
|  |  |
| --- | --- |
| **Final Term Project** |  |
| COMP6175  Object Oriented Programming |
| **Teknik Informatika (Master Track)**  *Computer Science* | **Anggota Kelompok**  *Members*  Jonathan Sebastian P. (2101626626)  Nicholas Dominic (2101628594)  Arvin Melvillo (2101628650)  Hendy (2101632225) |
| **09 Juni 2018**  *June 09th, 2018* |

**LIBRARY APP**

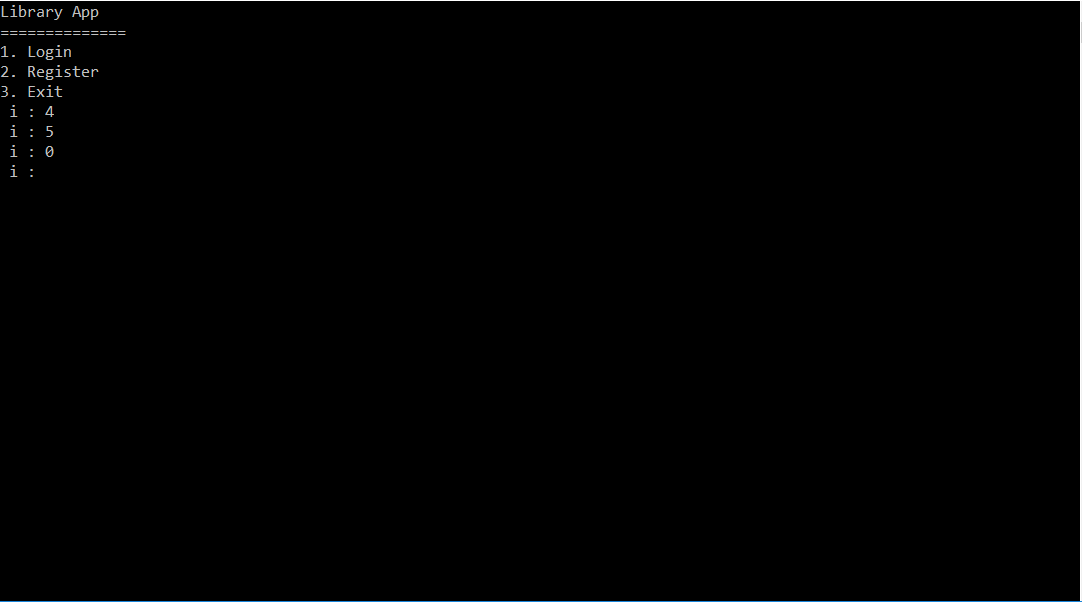


\*\* activities that’re outside the program

The library app is a program to help users (librarians/ readers) on processing datas, such as, see all listed available items, lending an item & returning an item with keeping trace of user, updating item’s stock, etc.

The program is command-line interfaced, needing input from its user to run its program.

**LOGIN/ REGISTER MAIN MENU**

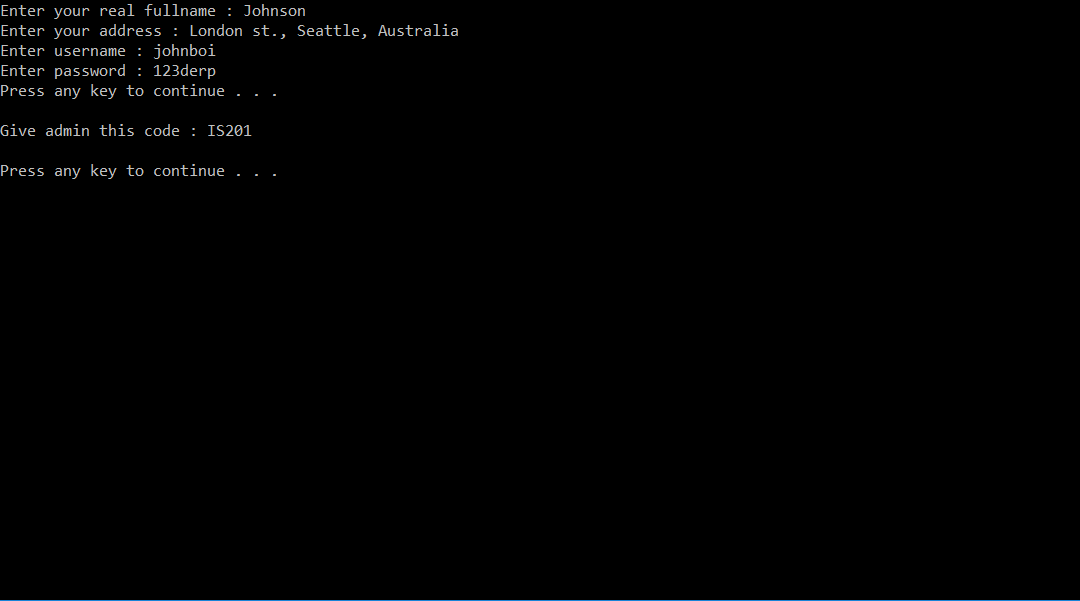
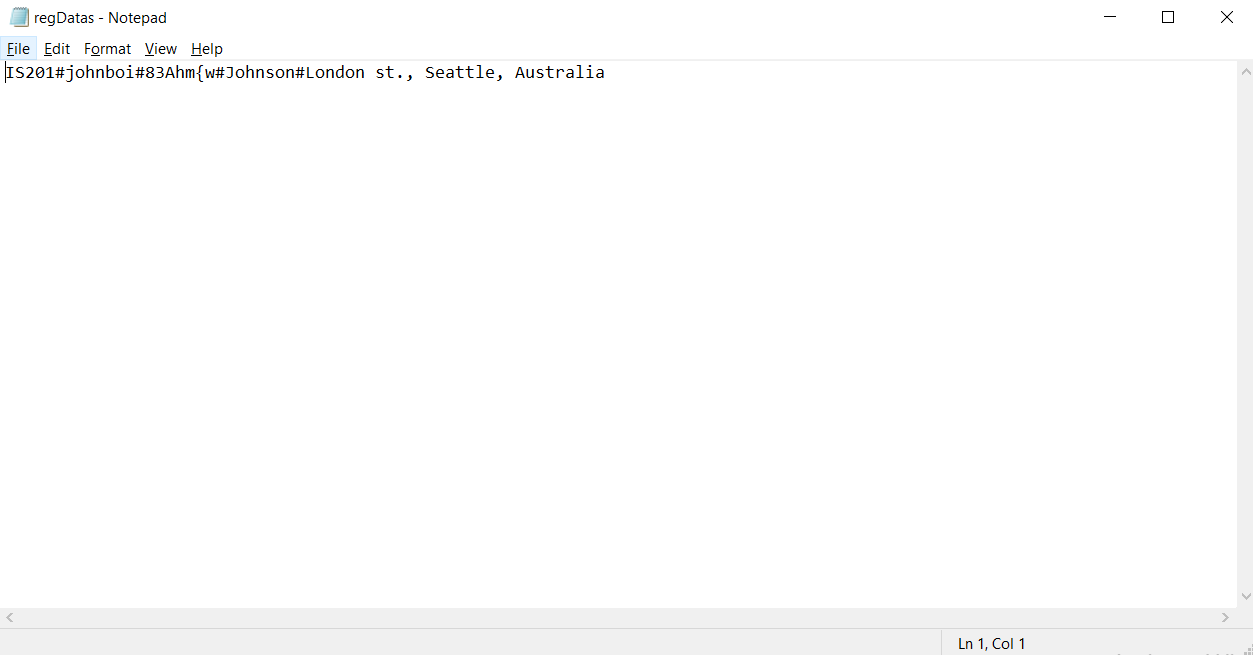


On opening the app, the login/ register menu will appear, asking either if user wants to login, register, or exit application. User input will be validated to make sure its input is within the program options.

**REGISTER MENU**



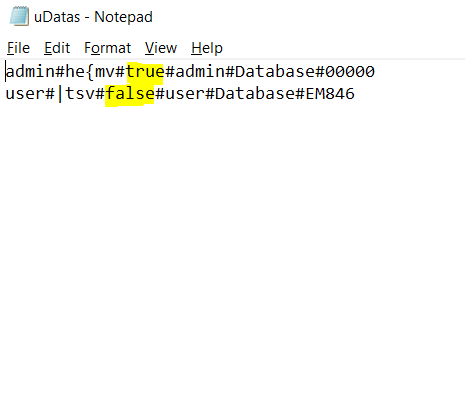
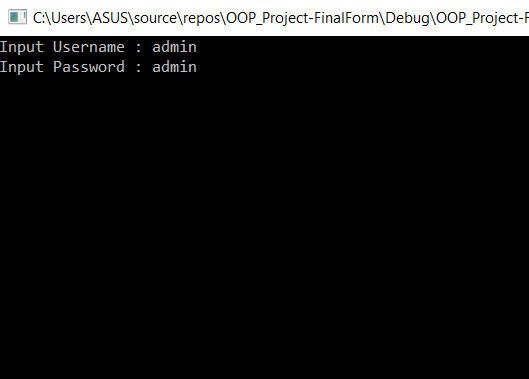
When opening the register menu, user will be asked to input his/her name, address, username, and password. After they enter all datas, they are **NOT** registered as a user yet.

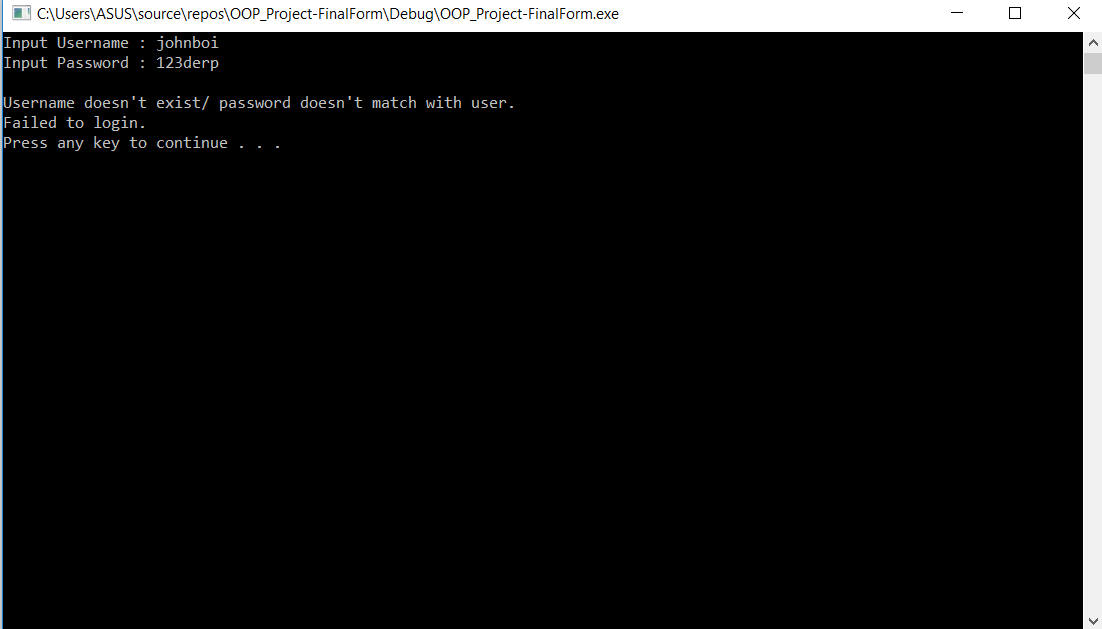
After they finish entered all data needed, they are **NOT** registered as a library user yet. Instead, they will be shown a random code , at which must be given to the librarian\*\* so the data can be validated by the librarian from the registree identity and the data inserted on the database to confirm registration. The data is saved on the **Register Database**.

After that, they will be returned to the **Login/ Register Main Menu.**

**LOGIN MENU**

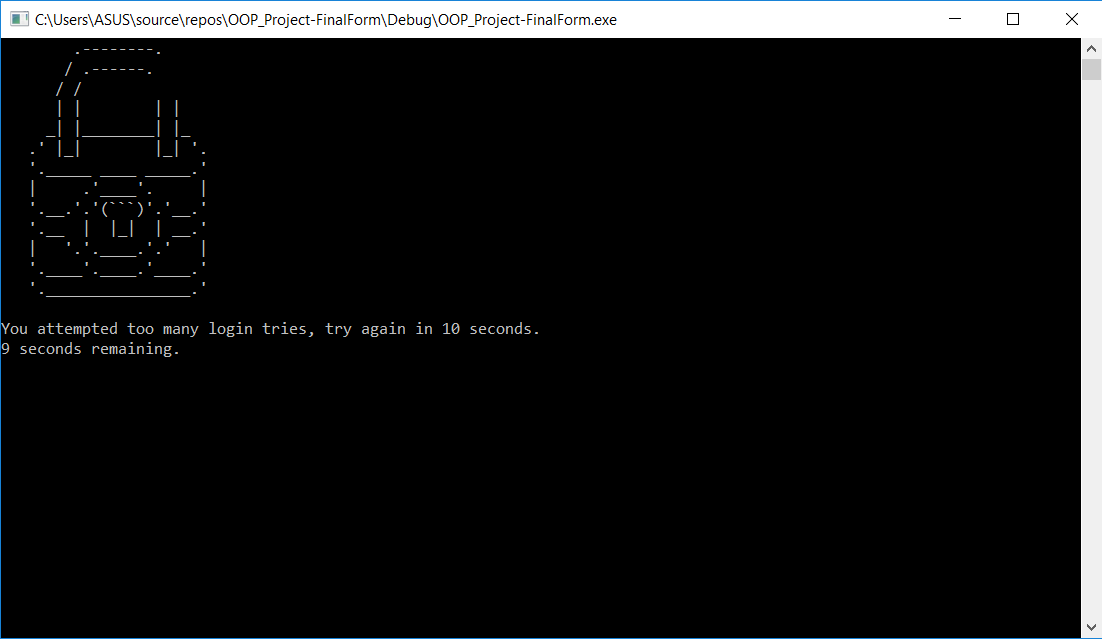
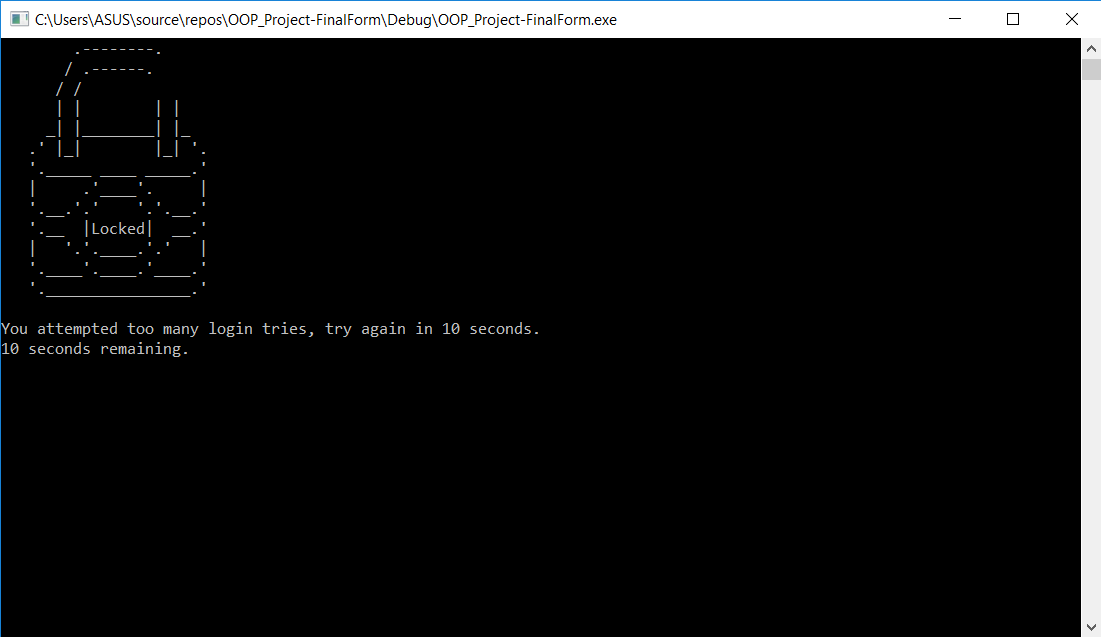
 

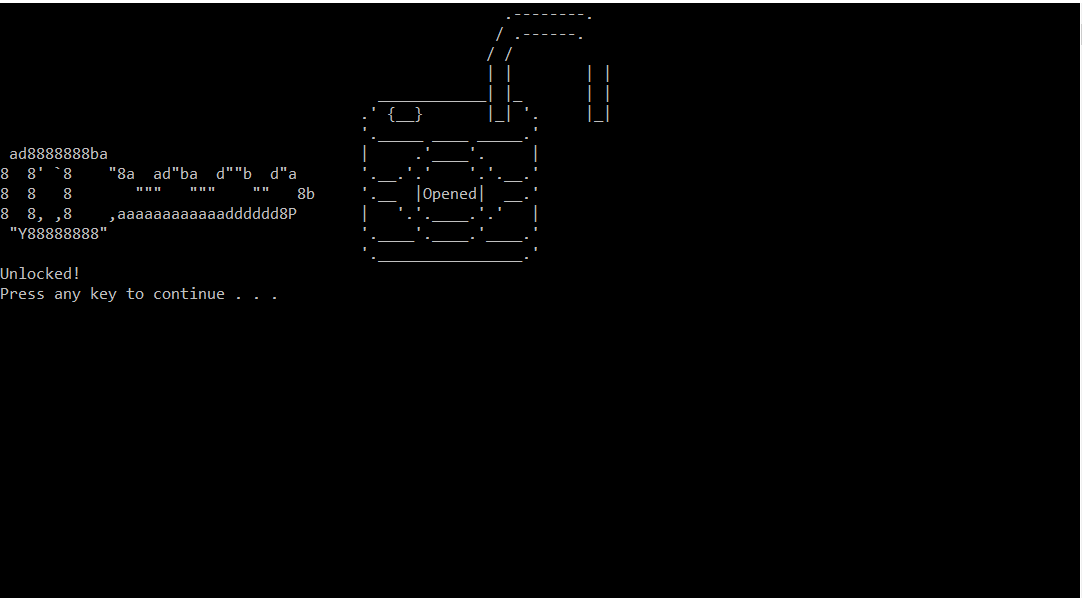
When opening login menu, the program will open the **User Database** and read all of the data and put each line as a class object. Each line in the database contain username, encrypted password, user type (highlighted), user real name, user address, and user ID code. Then, the program will ask the user to input his/her username and password, then match the username and decypted password with the data.



If the program fails to get a match from any of the user data, the program will return to **Login/ Register Main Menu**. However, keeping counter on how many times has the user failed to login. If the user attempted too much login without success, the program will run **Lock System**.

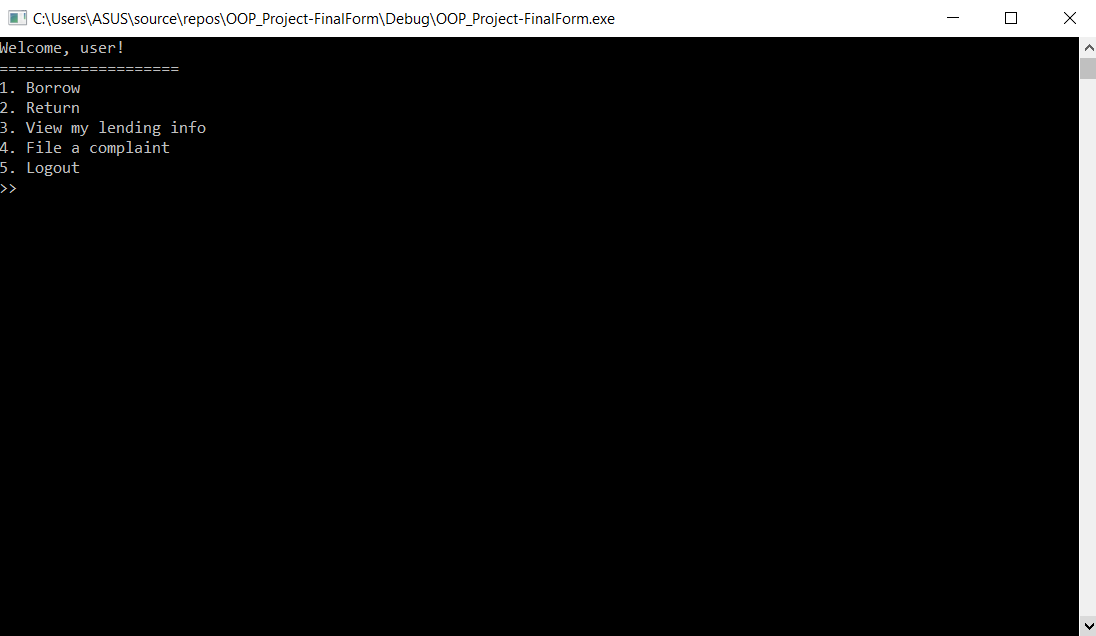
**LOCK SYSTEM**

If the user tried 5 login attempts without success, the program will lock itself down, while displaying a timer and a message. The user must wait until the program timer ran out. Then, the program will return to **Login/ Register Main Menu.**



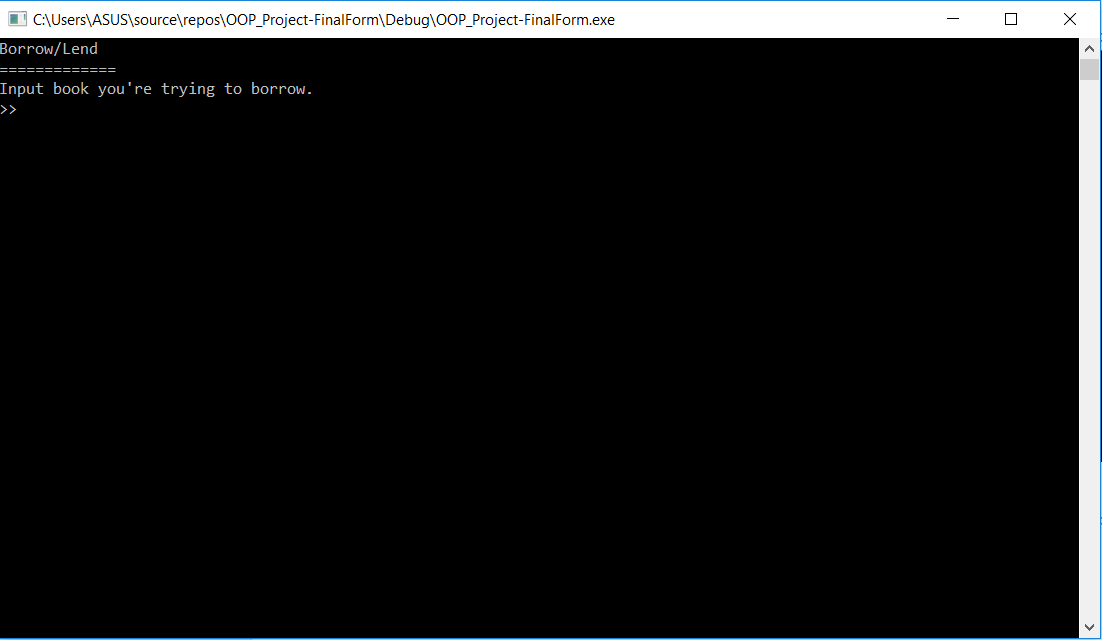
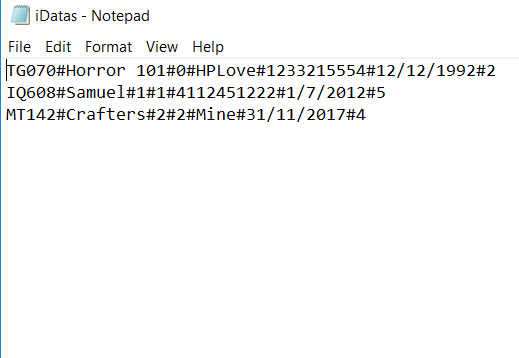
If the program found a match, however, the program will copy that user data as a new object as known as *This User* (to imply who is running the function, and easier access to save that user data when needed) and the program can do one of two things, open **Admin Interface Menu** or open **User Interface Menu**. The way the program know which interface menu it should open is by reading its user type in boolen form from database wether it’s an admin (true) or a user (false). the false value will open **User Interface Menu**, while true value will open the other.

**USER INTERFACE MENU**

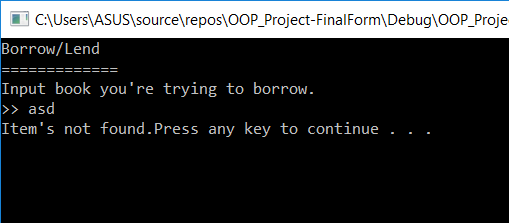
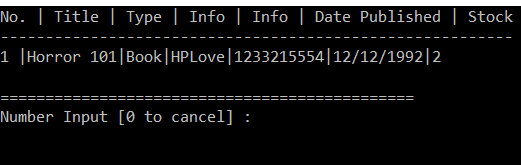


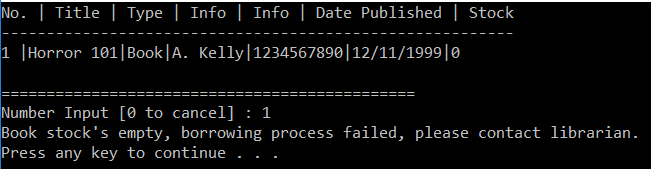
This menu consists of 5 options, **Borrow Option**, **Return Option**, **View My Lending**, **File Complaint Option**,Logout option. Logout simply returns to **Login/ Register Main** Menu. User will be asked to choose within these options. input will be validated.

**BORROW OPTION**

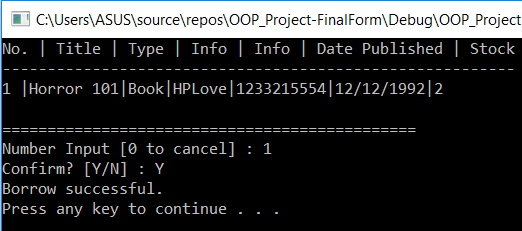


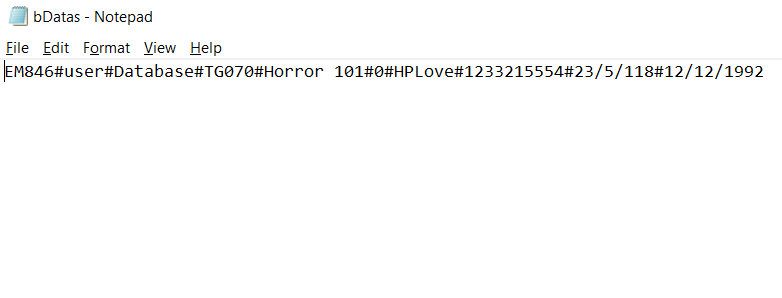
Upon opening the borrow option, program will read all data on the **Item Database**. Then user will be asked to input the item’s title, which then the program will try to find. The way how the program search is by **Search Each Word**.

If there’s no item found, program showed that there is no found item. Else, show every item that match input, and ask input on which item to borrow. 

After input on an item, the program will see if the stock is not empty. if its’ empty, the program will display the stock is empty.



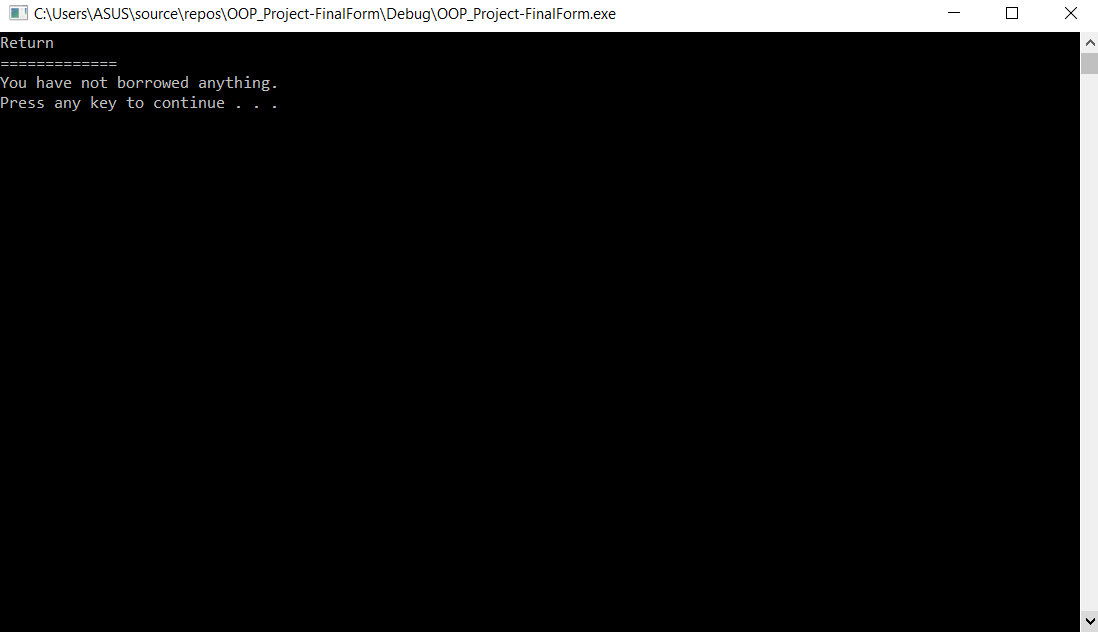
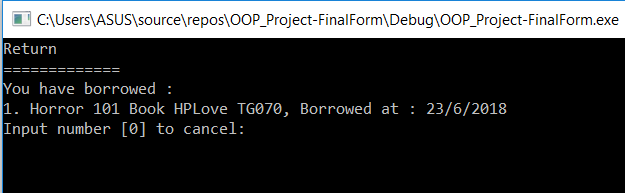


If it’s not empty, it’ll proceed to ask the user for confirmation, and when they do, the program will save *This User* info (ID code, name, address) info and the Item info (ID Code, title, info, other info, item published date, and borrowed date) on the **Borrowed Database**.

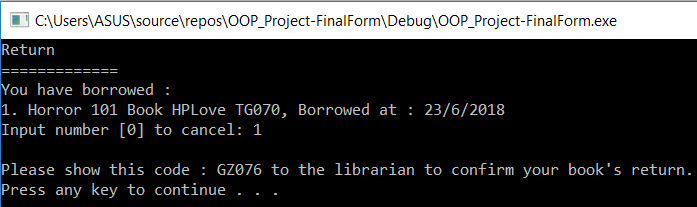
**SEARCH EACH WORD**

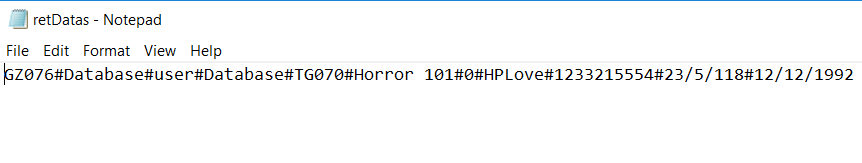
Searching function, where user input’s string will be compared with each word in each item. the program will try to compare each letter from the first letter of the user input and the first letter of the first item’s word (before a space) when comparing each letter, if a letter is not the same, it’ll flag it as not the item user inputted, and then search again on the second’s word of the title(after a space) and compare it with the input (until there’s no more space). This’ll loop until all item has been compared.

**RETURN OPTION**



When opening this option, the program will read every data from **Borrowed Database**, and compare *This User* ID code with each item’s borrower ID code. Then it’ll show every item that’s borrowed by *This User* (none if user doesn’t borrow anything). It’ll then ask input on the index number on which item the user would like to return.

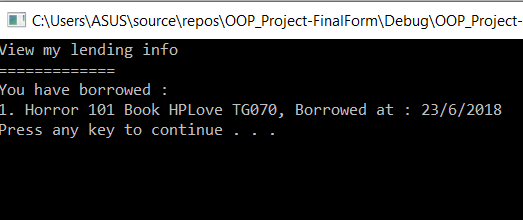




When entering a item the user would like to return, the user will be shown a random code that must be entered by the librarian to confirm it. This function does **NOT** mean the user has returned the item.

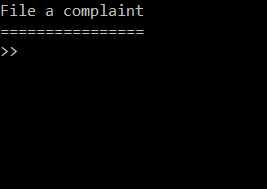
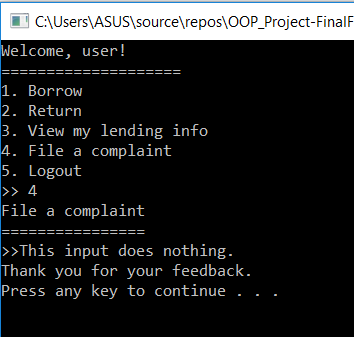
Data from the **Borrowed Database** will be copied to the **Return Database**.

**VIEW MY LENDING**



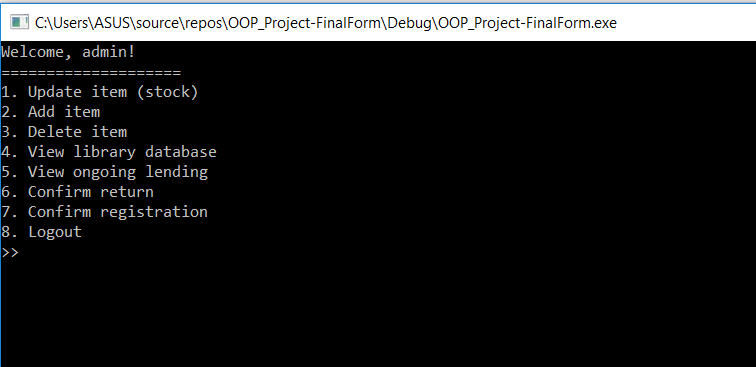
On this option, it’ll only show on every item which the user has borrowed, just like the return function.

**FILE A COMPLAINT**

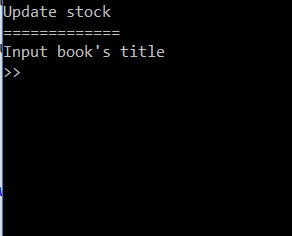
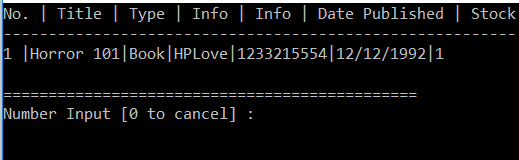
Is the user not satisfied with our very hard work of our services to them? That’s okay, we created an option specifically for the user, to vent on a machine rather than on somebody else. Sometimes user arrogance and pride can place themselves high enough where they feel that they can do annoying actions which are rich in toxicity, that it will affects other people that’re around him, such as, a librarian. That’s okay, we can understand that. That’s why, we created a solution, this option will ask for a user input. After input, the program will display a very polite, nice, and sweet thank you for the user. The input is not saved.

**ADMIN INTERFACE MENU**

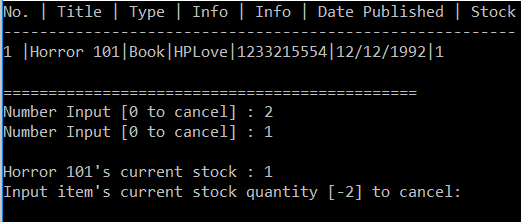


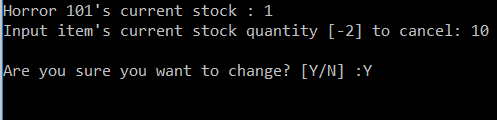
If the program see that the user is a admin (librarian), it’ll open this option and showed these options : **Update Stock**, **Add Item**, **Delete Item**, **View All Item**, **View All Lendings**, **Confirm Return**, **Confirm Registration**¸ and Logout Option. As usual, logout option will return user to **Login/ Register Main Menu**. Input is validated.

**UPDATE STOCK**

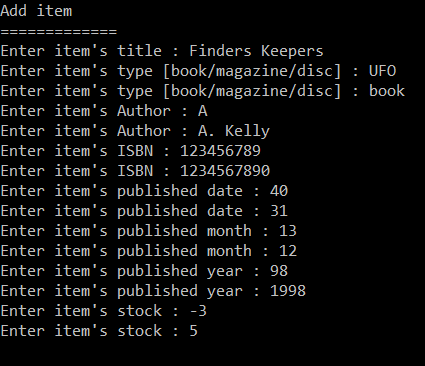
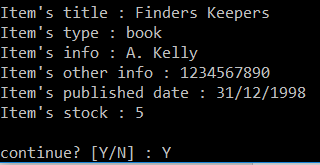
On updating stock option, the program will read every item on **Item Database** and ask for a user input. it’ll search with the **Search Each Word** function, and when found, it'll display all available items.



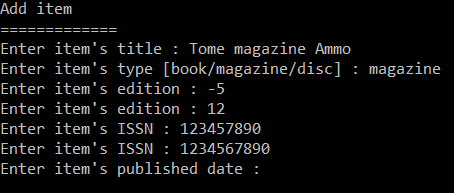
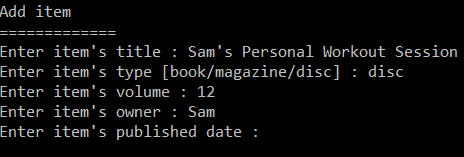


After choosing a item, it’ll display item’s current stock, then ask for the change of stock to user. After confirmation, the program will overwrite data on **Item Database**.

**ADD ITEM**

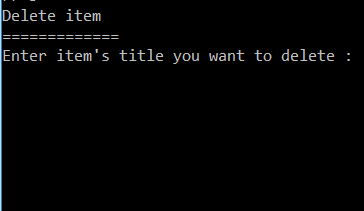
On adding an item option, it’ll ask for a title, type, info, other info, published date, and stock. All of which are validated. Author needs minimum input of 3 letters, ISBN info must have 10 letters exact, date must be in calendars format, and stock can’t have negative value.

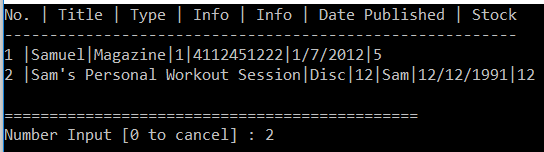
For every type, however these info are not the same (disc has volume and owner, magazine has edition and ISSN), since the program define type as different classes which inherited to another class as a same object (these parents only define the types, while the inherited class define other info, such as title, published dates, etc, to avoid diamonds problem).

After confirming on adding an item, it’ll add the item to the **Item Database**.

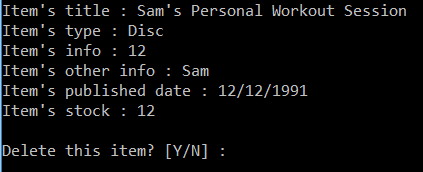
**DELETE ITEM**



On this option, the program will read all data from the **Item** Database. The user will be ask for input. This input will be search with **Search Each Word** function to find all available item from input.

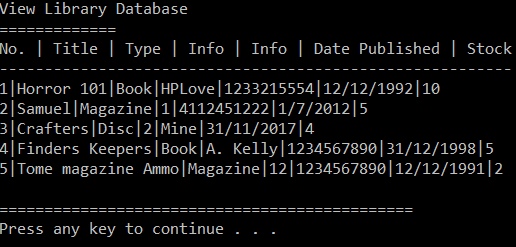


Same as usual, it’ll display all item based on input, and ask for more input on the item user wants to delete.



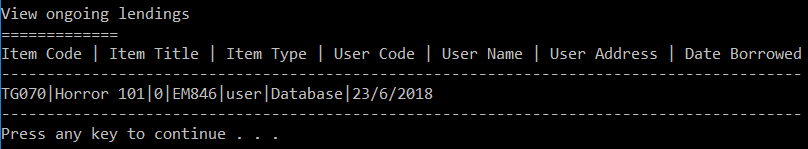
After entering a item’s number, it’ll display all its information and ask for a confirmation. On confirm, it’ll delete that specific object, overwrite all data to **Item’s Database**.

**VIEW ALL ITEM**



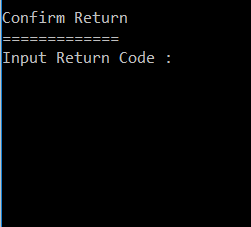
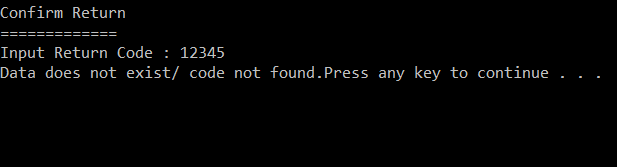
View all item simply display all items in the **Item Database**, and show all its information.

**VIEW ALL LENDINGS**



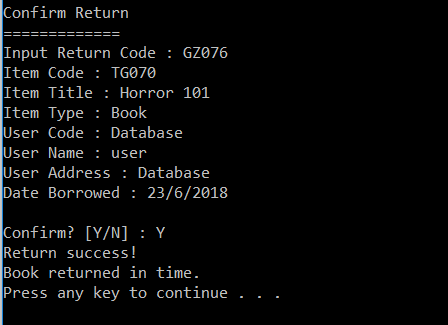
On view all ongoing lendings option, it’ll show all datas on **Borrowed Database**, showing all infomation from borrowers to the items.

**CONFIRM RETURN**

On confirm return option, it’ll read all datas on the **Return Database**, and ask for a user input.

The option will return the user to **Admin Interface Menu** when not found.

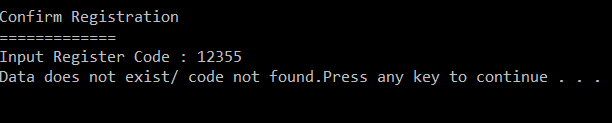


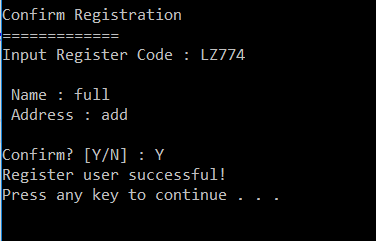
When found however, it will show all info about the item, and then ask for confirmation to return it.

The program then will decide if the return is late or not (we built it to have a 2 week deadline), if it’s late it’ll show the fine for the charges (we made it $1 a day).

If the item is found on **Item Database**, it’ll increase the stock by one, if not found, it’ll make a new item to the **Item Database**. It’ll delete the specific data on **Borrowed Database** and **Return Database**.

**CONFIRM REGISTRATION**





On confirm registration option, it’ll ask for the code that’s displayed when user registered. when found, it’ll show the registree information, and ask for confirmation. When confirmed, register the user to **User Database** (as a library user), and it’ll delete the data on **Register Database**. The registree has fully been registered.

**OOP APPLICATION**

How the OOP applied in this program, is based on the databases. There are 5 databases, and they are :

* **ITEM DATABASE**
* **USER DATABASE**
* **REGISTER DATABASE**
* **BORROWED DATABSE**
* **RETURN DATABASE**

All of these, have specific class, which on read will define its objects. They have the user-defined setter getter function and things. on the **Item Database**, we have built a three-way class that defines each type of an item (book has author & ISBN; magazine has edition & ISSN; disc has owner & volume) that will inherit to a core class (essentially, like a fork). We also define overloading in the code which cannot be seen in the program.