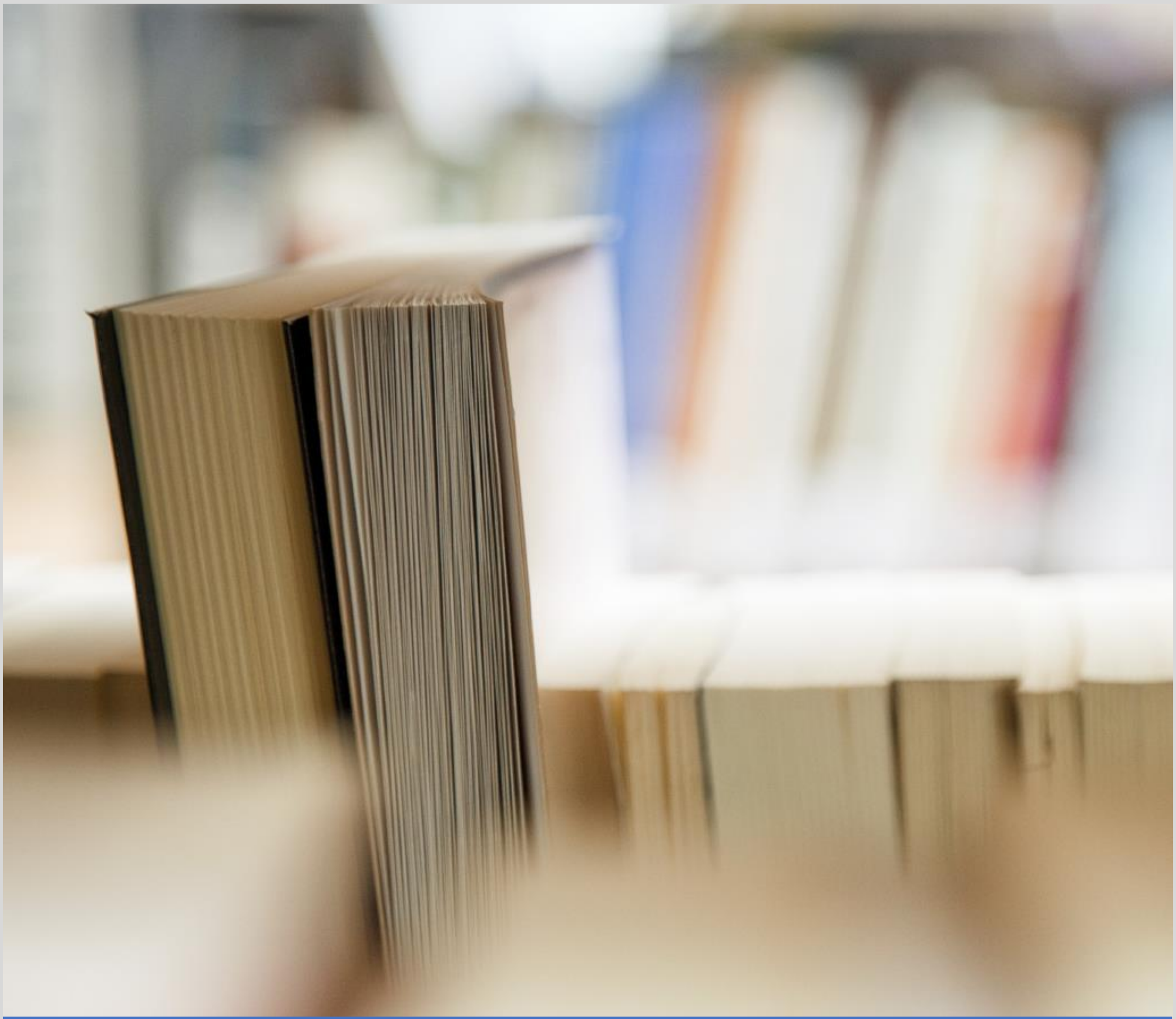


Library Management

C++ Final Project

Malekghasemi, Amirali

4/6/21

Computer Science

Introduction

Design Overview:

The program Library Management is designed for efficient and easy account and book management by the library manager, employer, or organizer. User can easily add accounts and books to the database, control the borrowed and returned books and view information about the books and borrowed books through the program.

There are 5 main classes in the program, 3 of them are classes which represent different type of the accounts (named NormalAcc, PremiumAcc, Luxuryacc) and they are derived classes of the base class Account which contains the common functions and variables that these derived classes use. There is a class for books called Book which contains all the functions and information related to the books of the library.

Every aspect of the program is done as an object of one of the 3 account classes or the book class. The account that user adds is an object assigned to a different pointer in an array of pointers related to that specific type of the account (Normal, premium or luxury). Same goes for each book that is added with the difference that all books are stored in only one common array of pointer which holds the objects of class Book. Each account can be created, assigned to borrow a book(Depending on the account the number of books allowed to be borrowed at once changes), return a book, and show the info of their borrowed books. Books can be added and the list of all the books in the library can also be shown at once. In doing of all these operations, the 3 account classes and Book class go hand in hand and work simultaneously to make the operation possible.

At the start of the program user may choose one of 7 options. Each option is assigned to a specific case in a switch statement which contains specific functions related to that option. By inputting the right integer user will be transferred to the proper page and shown instructions for further input. Any wrong input will be attended to by an exception or and if-statement and will be shown the appropriate error message.

Due to the input sensitive nature of the program, there are many if-statements embedded to each option to find the right output for the user. The program contains many validation processes and contains many user-friendly and visible outputs to furthermore improve the user-friendly interface.

SYSTEM ARCHITECTURAL DESIGN

Chosen System Architecture: Layered pattern

Main program:

- Contains 4 array of object pointers, 3 for different accounts and 1 for books.
- Class Account has all the setters and getters that are used in the three derived classes of it(NormalAcc, PremiumAcc, LuxuryAcc). It also stores the important variables of each account in its private section which are first name, last name and id number.
- The program uses while loop to iterate until user choses to exit and stop the iteration.
- Depending on user's input, the switch statement within the while loop directs the user to their piece of operation and code.
    - User can select one of 7 options to run.
    - By choosing 1 or 2 users can add a new account or/and a new book. The information for these two will be stored as objects pointers in their respective array of pointer.
        - If user creates an account with the type normal, an object will be created and assigned to one the pointers in NormalAcc array of pointers. Normal accounts are allowed to borrow up to 3 books at a time, although this can change by simply changing the static value booksAllowed.
        - If user creates an account with the type premium, an object will be created and assigned to one the pointers in PremiumAcc array of pointers. Premium accounts are allowed to borrow up to 5 books at a time, although this can change by simply changing the static value booksAllowed.
        - If user creates an account with the type luxury, an object will be created and assigned to one the pointers in LuxuryAcc array of pointers. Luxury accounts are allowed to borrow up to 10 books at a time, although this can change by simply changing the static value booksAllowed.
        - If user adds a book an object will be created and assigned to one the pointers in Book array of pointers.
    - Options 3 to 6 are all done with the usage of information and functions in both book and accounts classes. User can borrow a book, return a book, see the list of borrowed books by one specific account or see the list of all

    books. All these are done by matching the id number of an account with the borrowedBy variable stored in Book class.
  - By choosing option 7 user can end the program. By doing this all the information stored in the program will be lost
- All the user inputs are validated during the process of all options using specific if-statement that use classes of exceptions to output an error message.

Layered pattern:

This program is divided into 6 subtasks. 4 of these subtasks need the other two subtasks. How this works is specified below:

1. Layer 1 is getting the input from the user to create accounts and books.
2. Using layer 1, layer 2 then can operate its tasks with the info it gets from layer 1 for example for borrowing a book, you need the id number of the account and the number of the book which was created in layer 1.

Alternative Designs:

At first my original plan was to have a pre-established database containing all the book and account's information. Then from there we could add new data to that database. Unfortunately, this was not possible due to lack of knowledge and the fact that storing data would need a specified space on a server or hard drive. One of the other cons of this design is finding the right format to store the dataset. For example, I was thinking about storing data in an open XML file but doing this would defeat the whole purpose of using subclasses. This also means that everyone can access the data by copying the files unless I encrypted them. The pros of this design are that the inputted data will not be lost anymore after the program has been terminated, where as now, the inputted data is stored in a temporary memory.

One of the other design plans was having every switch case as a function. This meant that instead of the codes related to each case in int main() have them in a function defined outside main. The pro of this would be easy traceability and having a more organized code. Although this would mean more complex cross calls between function and classes. I did not go for this design because each variable and each class are related closely, and each option of the program uses all of them at the same time. I did not want to sacrifice functionality over looks.

The best design for the program would be having this code with object pointers storing data in an external server or database. This would tackle the current issue of this program which is loss of data after it is terminated. This will double the complexity of the code and would need much more time and knowledge.

<u>Detail description of components:</u>

Components:

Classes:

- Base class Account:
  - Base class for 3 classes : NormalAcc, PremiumAcc, LuxuryAcc
  - Contain all the setter and getter functions related to creating an account
  - Contains the variables responsible for storing each account important information(e.g. first name)
- Subclass NormalAcc
  - Calls to setters function in Account from its constructor to store new data
  - Uses function and variables form Base class Account
  - Has one independent static variable which differentiates it from the other classes(number of borrowed books allowed)
- Subclass PremiumAcc
  - Calls to setters function in Account from its constructor to store new data
  - Uses function and variables form Base class Account
  - Has one independent static variable which differentiates it from the other classes(number of borrowed books allowed)
- Subclass LuxuryAcc
  - Calls to setters function in Account from its constructor to store new data
  - Uses function and variables form Base class Account
  - Has one independent static variable which differentiates it from the other classes(number of borrowed books allowed)
- Class Book
  - Responsible for storing data about books
  - Has all the functions related to books
  - Holds all the data about each book in its private variables

- AccTypeException, IdException and BookException classes
    - Subclasses of superclass exception
    - Get called upon to throw an exception in case of invalid input
    - Contain the error message related to the invalid input

Functions:

- Introduction()
    - Prints out the details about the program
    - Prints out a welcome message to improve user-interface
- Menu()
    - Print out the menu and keeps printing after each operation until the user stops the program
    - Has a user-input validation embedded which throws an instance of invalid_argument if the user’s input is wrong
- Functions in Base class Account
    - Has setters and getters for:
        - First Name
            - void setFirstName()
            - string getFirstName()
        - Last Name
            - void setLastName()
            - string getLastName()
        - Id number
            - void setID()
            - string getID()
    - These functions whether set the private variables to user’s input or get these private variables and return them as public variables so that they can be used in int main()
    - Has a function for increasing and decreasing the number of borrowed books of each account:

        - Every time an account borrows a book their number of borrowed books increase by this function
            - void incBorrowedBooks()
        - Every time an account returns a book their number of borrowed books decrease by this function
            - void decBorrowedBooks()
    - Has a function to get the number of borrowed books for each account from private variable and return it as a public variable so it can be used in int main()
        - int getBorrowedBooks()

- Functions in classes NormalAcc, PremiumAcc and LuxuryAcc
    - Each of these classes has a get function for:
        - gets the number of borrowed books allowed for each account from the private variable and returns it as a public variable so it can be used in int main()
            - int getBooksAllowed()
- Functions in class Book
    - Setter and getter functions for:
        - Book Name
            - void setBookName()
            - string getBookName()
        - Book Number
            - void setBookNumber()
            - int getBookNumber()
        - The id of the person that has borrowed the book
            - setBorrowedBy()
            - string getBorrowedBy()

    - A function to increase the number of total books every time a book is added and another function to get that variable and return it as a public variable so it's accessible from int main()
        - static void setNumOfBooks()
        - static int getNumOfBooks()
    - A function used only option 6 of the program which displays each books name, number and whether if it's borrowed or not
        - void info()

Pointers:

- Array of pointers which stores objects of the class NormalAcc
    - NormalAcc * nAccounts[maxNumOfAccounts]
- Array of pointers which stores objects of the class PremiumAcc
    - PremiumAcc * pAccounts[maxNumOfAccounts]
- Array of pointers which stores objects of the class LuxuryAcc
    - LuxuryAcc * lAccounts[maxNumOfAccounts]
- Array of pointers which stores objects of the class Book
    - Book * books[maxNumOfBooks]

If-statements:

- If-statements are used throughout the whole program
- Their main priority is validating user's input and guiding the program to show the right message for different tasks
    - For example if an account has exceeded the number of books it can borrow, if-statements will get involved and guide the program to show the user the right message.
- In int main():
    - Choice 1: If-statement checks for the account type and if it's wrong it throws an exception
    - Choice 3: If-statement check for the right id and book number and if any of these are wrong it throws an exception. In addition it also checks to see

    that the entered id has not exceeded the maximum number of borrowed books and if it has it prints out an error telling the user this.
  - Choice 4, 5 and 6: If-statement check for the right id and book number and if any of these are wrong it throws an exception.

Switch-Statement:

- Depending on user's input runs the right case(From 1 - 7)
- It can terminate the program

Variables:

- Int menu() function:
  - Bool inputValidation: Will stay true until user inputs the right number
  - String choice: string version of choice that stores user's input. It this was an int, user inputting anything except numbers would have crashed the program
  - Int choiceInt: If the string version is all numbers it will be transformed into int by stoi() but if it's not all number it will throw and exception
- Class Account:
  - Private variables
    - String firsName: Stores first name
    - String lastName: Stores last name
    - String id: Stores the id number
    - Int borrowedBooks: Stores the number of borrowed books by the account
- Classes NormalAcc, PremiumAcc and LuxuryAcc
  - All have private static functions which stay the same for all objects
    - static int booksAllowed
- Class Book
  - String bookName: Stores the name of the book
  - Int bookNumber: Stores the number of the book

    - Int borrowedBy: If the book has been borrowed this will store the id of the borrower but by default it is equal to -1 which means the book is not borrowed
    - Static int numOfBooks: How many books are in the library
- Int main():
    - Int choice: Stores user's input for which operation he/she wants to do
    - Bool runProgram: Stays true until the user decides to close the program and end the main while loop
    - Int maxNumOfAccounts: Stores the number of maximum accounts the code can have which in this case is 10000
    - Int maxNumOfBooks: Stores the number of maximum books the code can have which in this case is 10000
    - Int index: Stores the index in each case temporarily so that the program can access the right pointer at the right time
    - String fName: Stores the user's input for first name and passes it to class constructor
    - String lName: Stores the user's input for last name and passes it to class constructor
    - String accType: Stores the type of the account
    - String bookName: Stores the user's input for book name and passes it to class constructor
    - Int bookNumber: Stores the user's input for book number and matches it with the one stored in book pointer to do functions such as borrowing
    - String id: Stores the user's input for id number and matches it with the one stored in account pointer to do functions such as borrowing
    - Bool bookFound: If the book is found this will turn true and initialize the correct code
    - Bool bookFound: If the account is found this will turn true and initialize the correct code

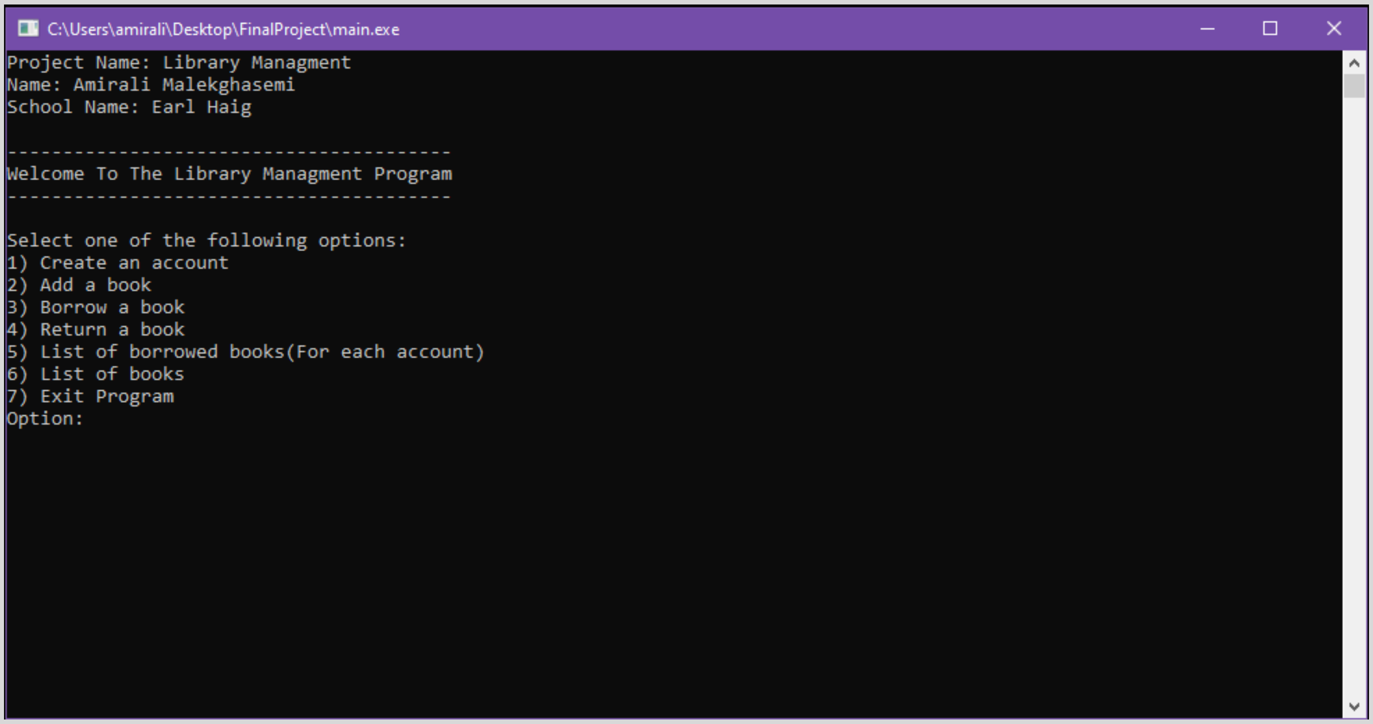
Loops:

- For-loops used in most of the cases
- For-loops used to search through all the possible indexes for the pointer in question
- While loop used in int menu() for repeating the menu if the user’s input is wrong
- While loop used in int main() to run the program as long as the user wants and exiting if the user choses to do so

Try and Catches:

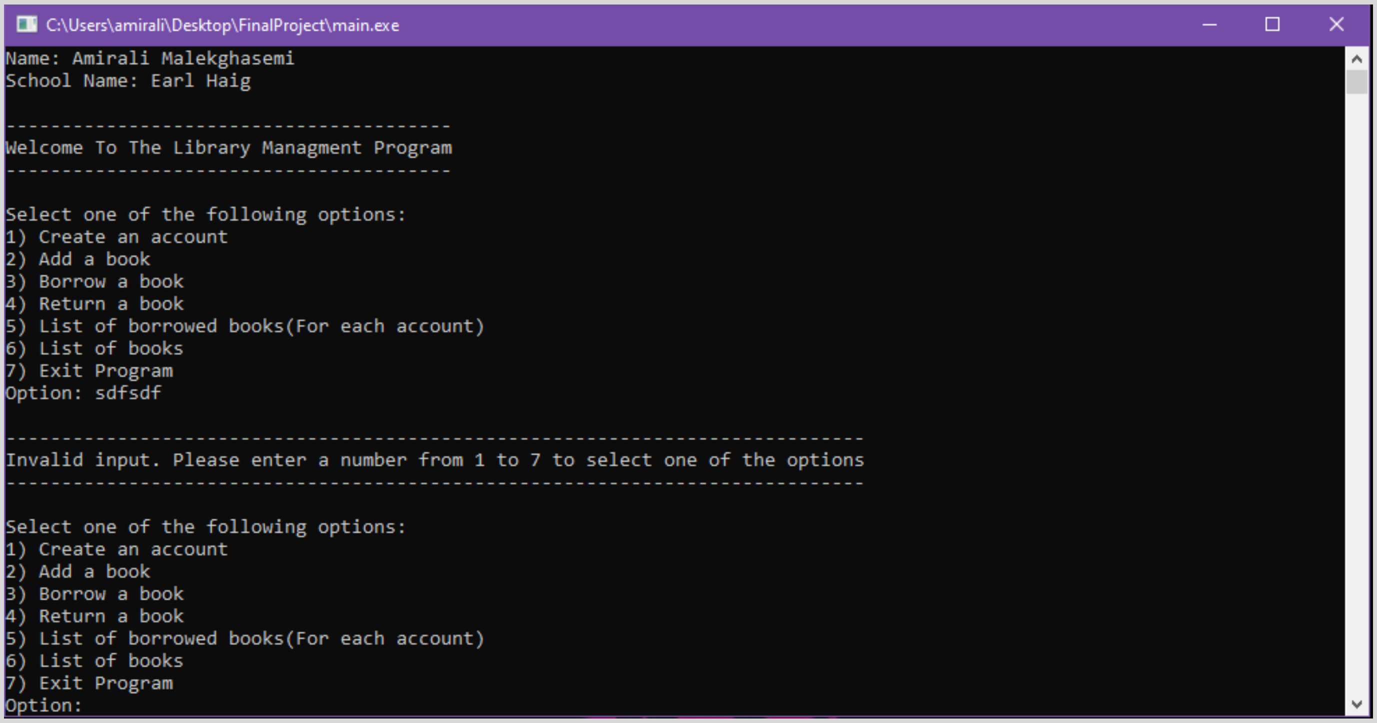
- Used through every single input
- If user’s input is invalid an instance of exception will be thrown from one of the exception classes and caught in the respective case

User Interface Design:



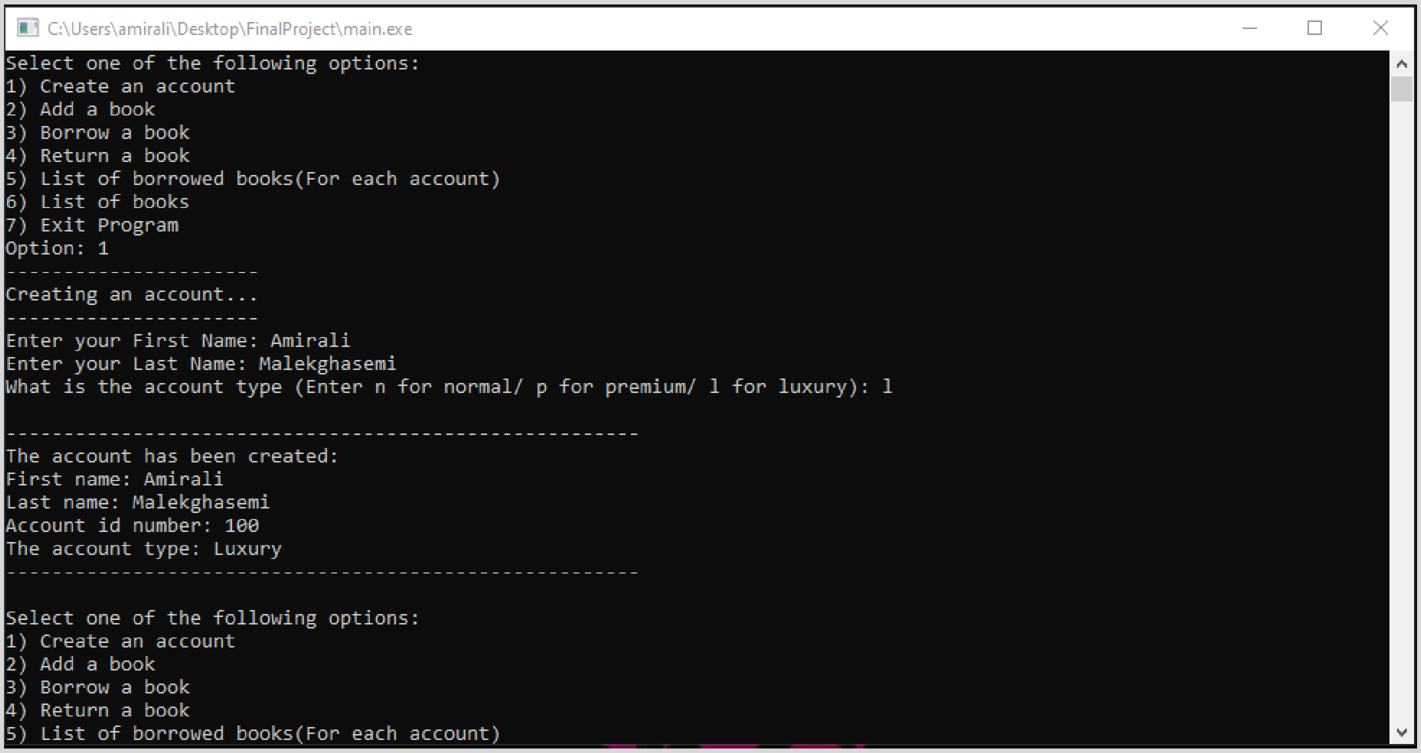
```
C:\Users\amirali\Desktop\FinalProject\main.exe
Project Name: Library Managment
Name: Amirali Malekghasemi
School Name: Earl Haig

----------------------------------------
Welcome To The Library Managment Program
----------------------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option:
```

Figure 1: The main page of the program and the welcome message



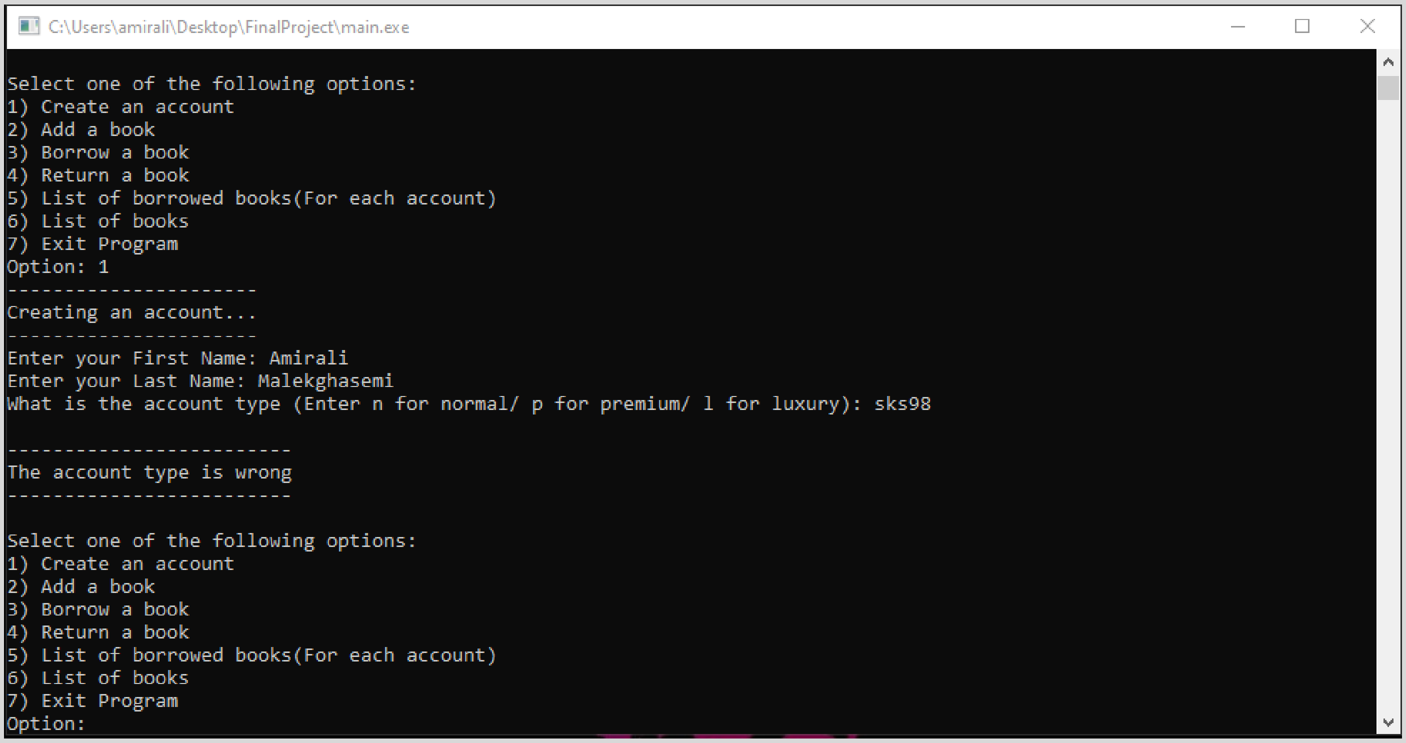
```
C:\Users\amirali\Desktop\FinalProject\main.exe
Name: Amirali Malekghasemi
School Name: Earl Haig

----------------------------------------
Welcome To The Library Managment Program
----------------------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option: sdfsdf

-----------------------------------------------------------------------------
Invalid input. Please enter a number from 1 to 7 to select one of the options
-----------------------------------------------------------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option:
```

Figure 2: Invalid input for function selection which results in an error message telling the user to enter the right input



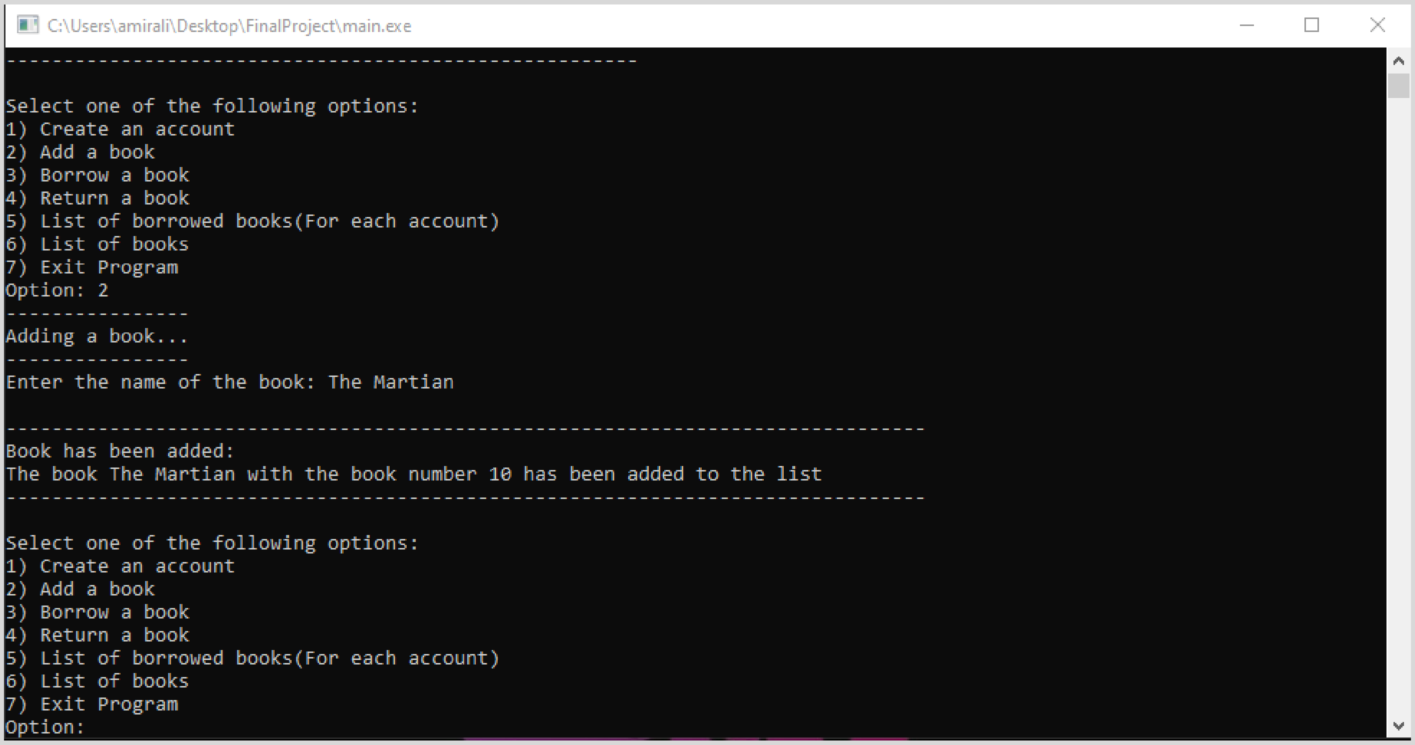
```
Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option: 1
----------------------
Creating an account...
----------------------
Enter your First Name: Amirali
Enter your Last Name: Malekghasemi
What is the account type (Enter n for normal/ p for premium/ l for luxury): l

-------------------------------------------------------
The account has been created:
First name: Amirali
Last name: Malekghasemi
Account id number: 100
The account type: Luxury
-------------------------------------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
```

Figure 3: Selecting option 1 and creating the account. First Name and Last Name do not have any validation process because it can differ in each country. The only validation is for account type. Here the account type is entered correctly, and the program has printed out the account information. Wrong account type can result in an error which is shown below.



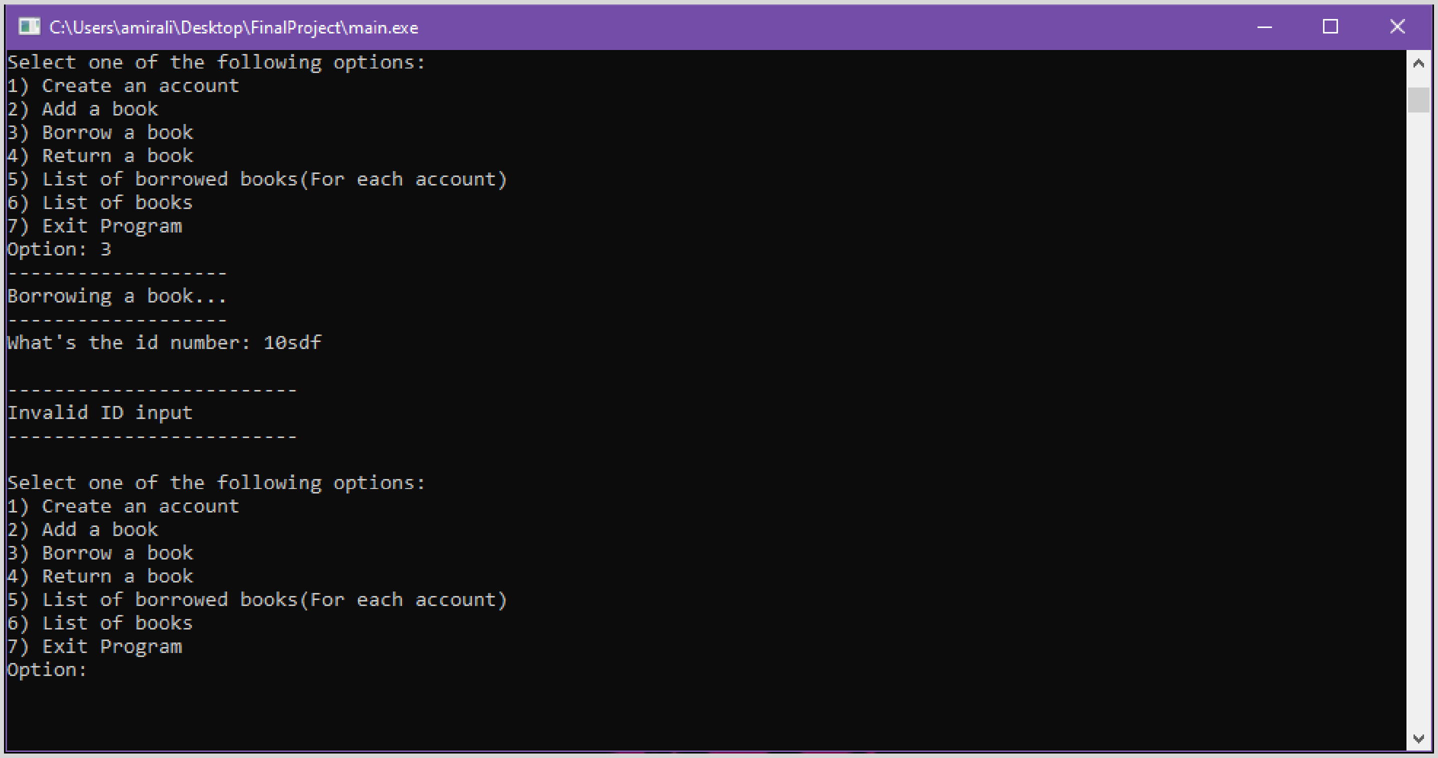
```

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option: 1
----------------------
Creating an account...
----------------------
Enter your First Name: Amirali
Enter your Last Name: Malekghasemi
What is the account type (Enter n for normal/ p for premium/ l for luxury): sks98

-------------------------
The account type is wrong
-------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option:
```

Figure 4: Error message outputted for wrong account type entry



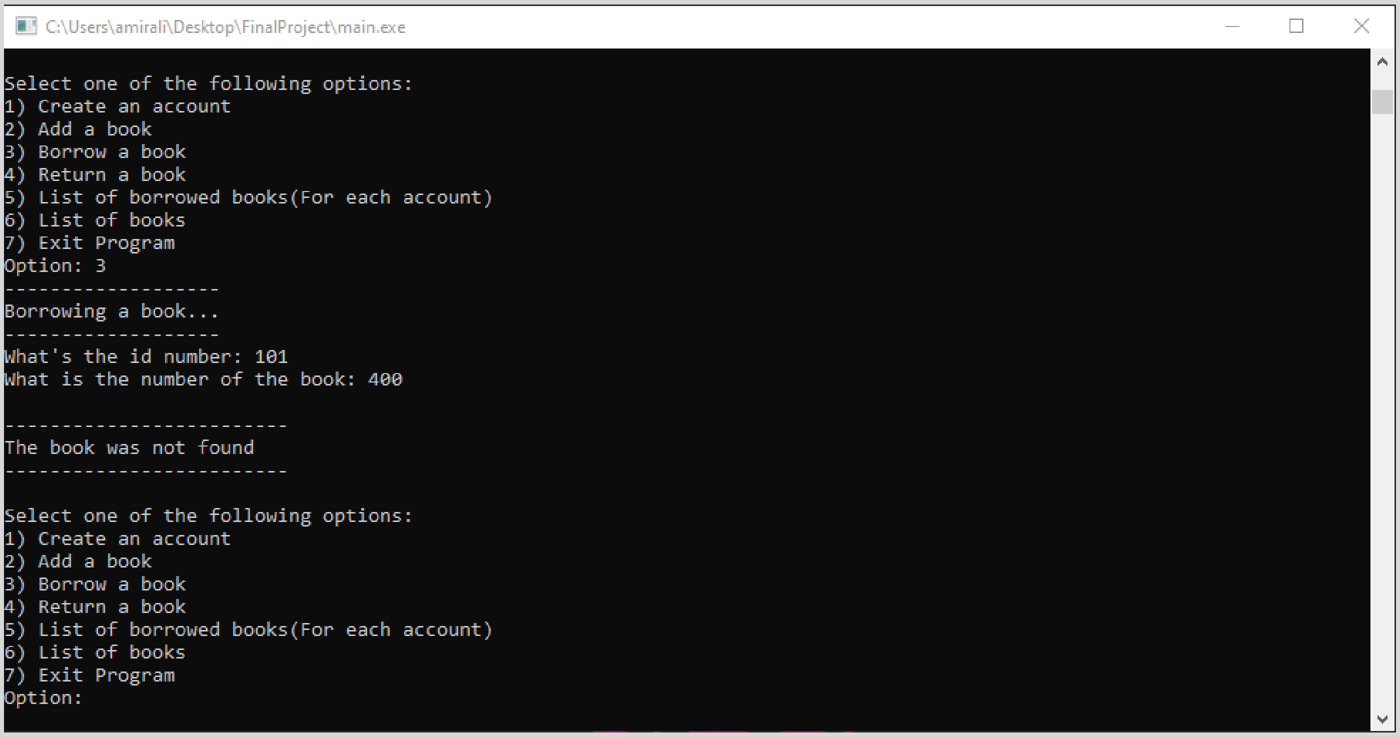
```
C:\Users\amirali\Desktop\FinalProject\main.exe
-------------------------------------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option: 2
----------------
Adding a book...
----------------
Enter the name of the book: The Martian

--------------------------------------------------------------------------------
Book has been added:
The book The Martian with the book number 10 has been added to the list
--------------------------------------------------------------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option:
```

Figure 5: Selectin option 2 and adding a book. Again book name doesn't have any validation process and after the book is made the info about it is outputted



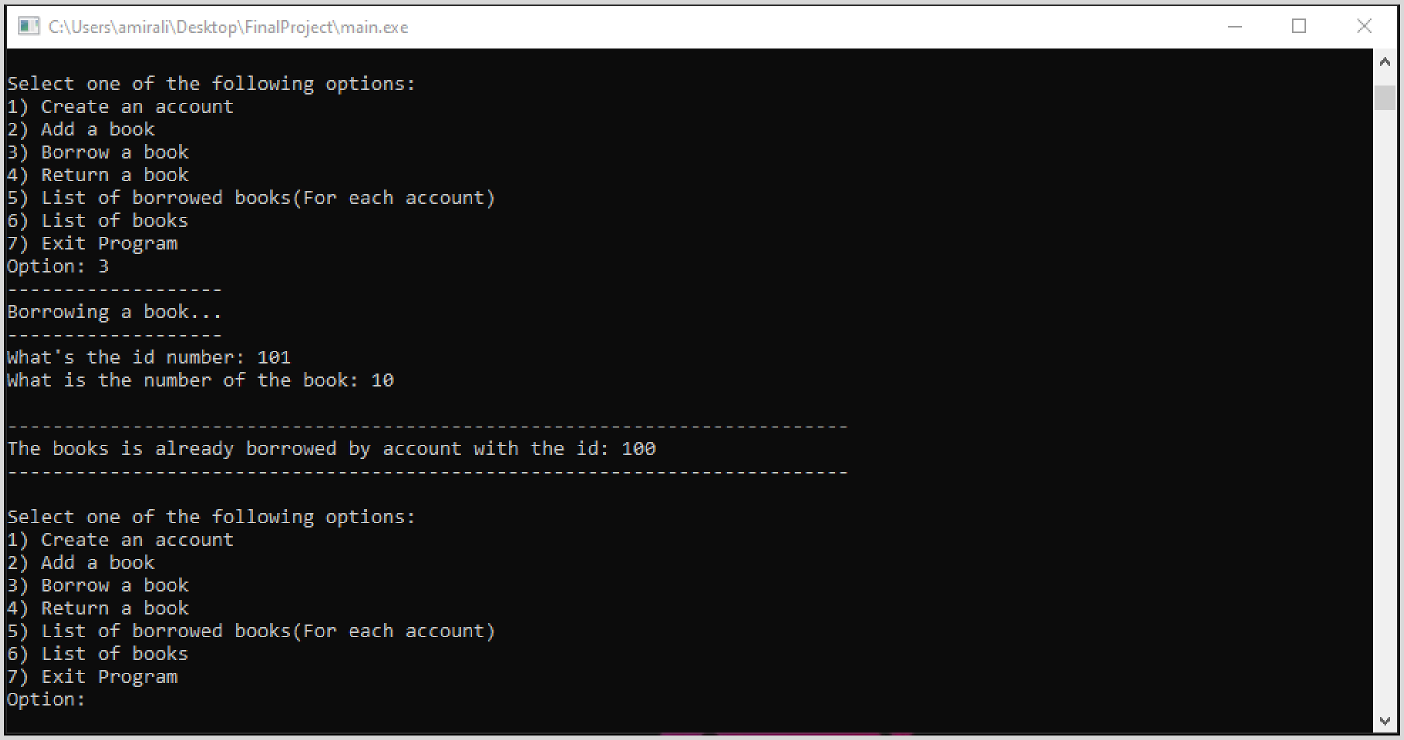
```
C:\Users\amirali\Desktop\FinalProject\main.exe
Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option: 3
-------------------
Borrowing a book...
-------------------
What's the id number: 10sdf

-------------------------
Invalid ID input
-------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option:
```

Figure 6: Wrong Id number input in any of the options will result in this error message



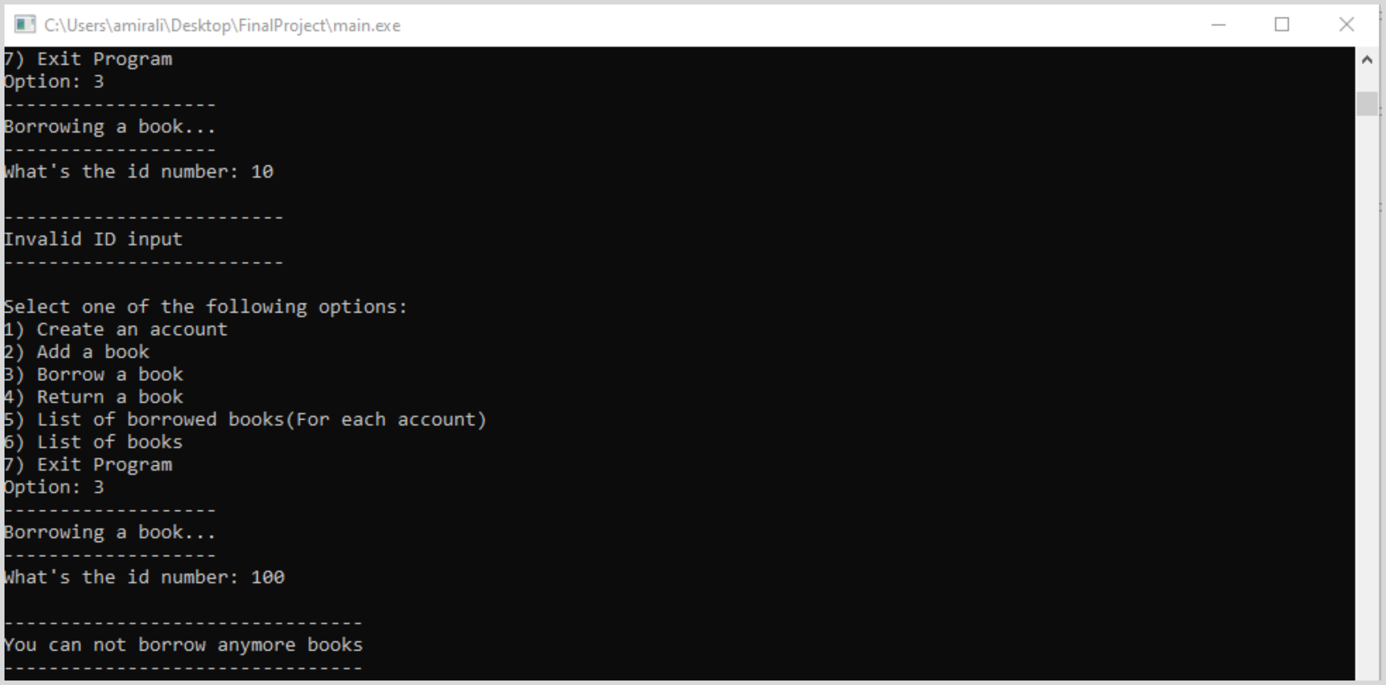
```
Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option: 3
-------------------
Borrowing a book...
-------------------
What's the id number: 101
What is the number of the book: 400

-------------------------
The book was not found
-------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option:
```

Figure 7: Wrong book number input in any of the options will result in this error message



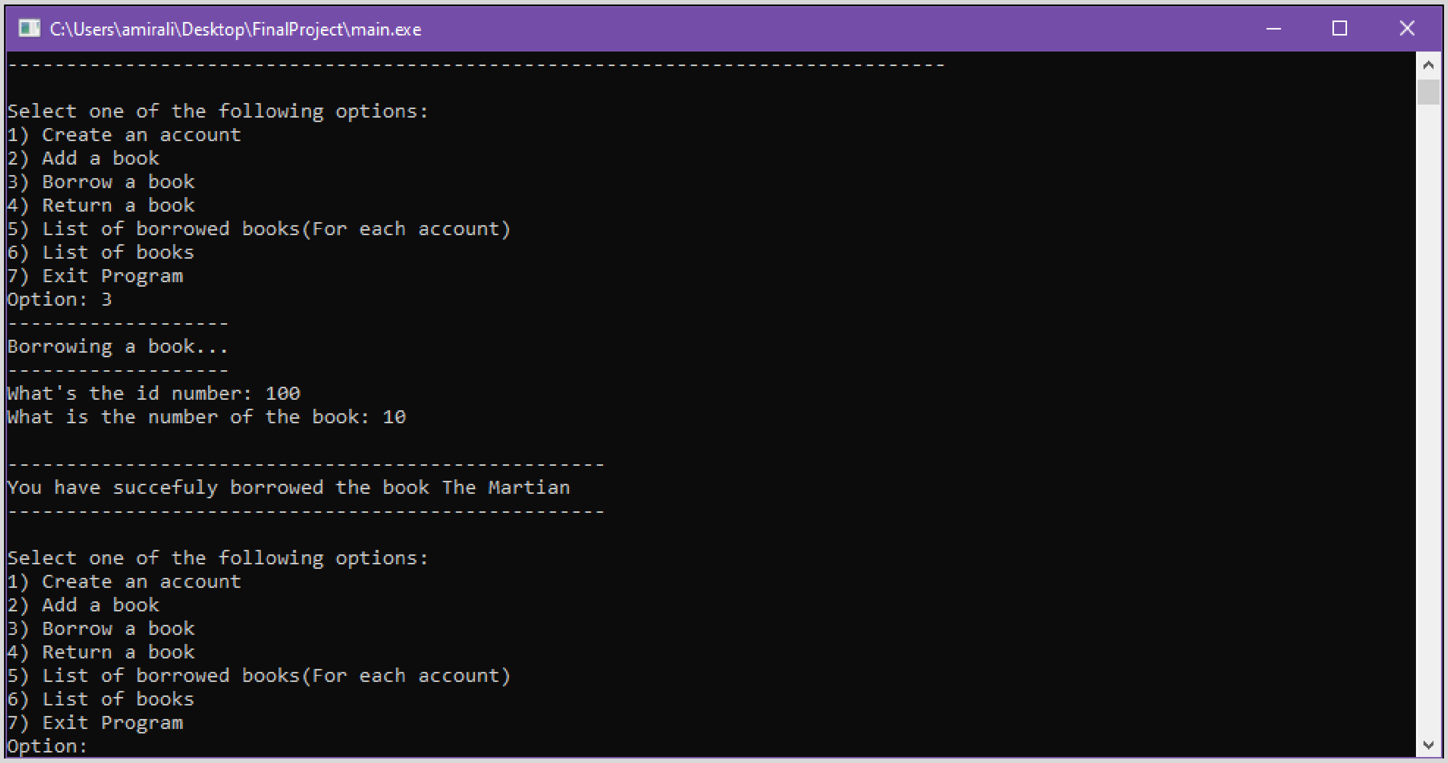
```
Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option: 3
-------------------
Borrowing a book...
-------------------
What's the id number: 101
What is the number of the book: 10

--------------------------------------------------------------------------
The books is already borrowed by account with the id: 100
--------------------------------------------------------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option:
```

Figure 8: If in option 3, both account and book numbers are correct, but the book is already borrowed, this message will be outputted informing the user about the situation



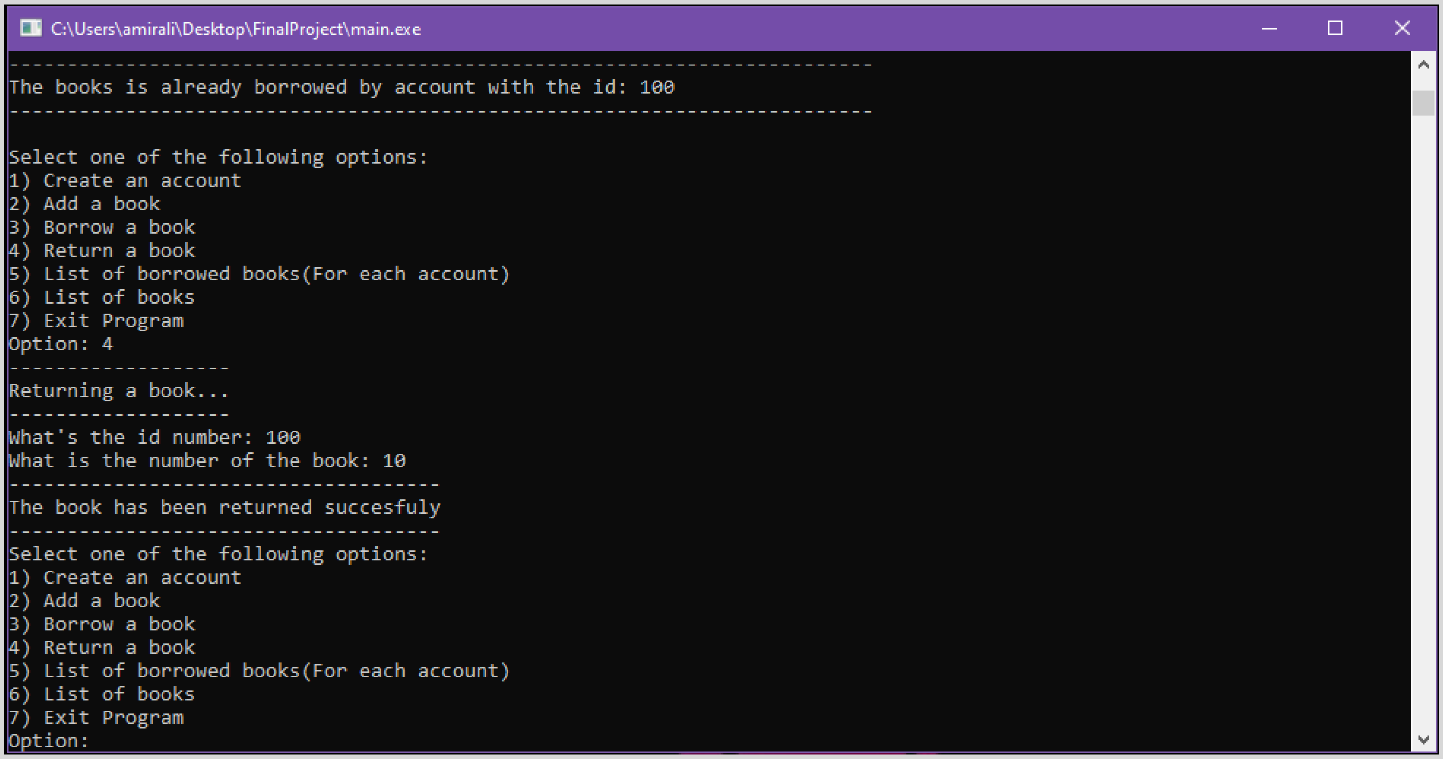
```
C:\Users\amirali\Desktop\FinalProject\main.exe
7) Exit Program
Option: 3
------------------
Borrowing a book...
------------------
What's the id number: 10

-------------------------
Invalid ID input
-------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option: 3
------------------
Borrowing a book...
------------------
What's the id number: 100

--------------------------------
You can not borrow anymore books
--------------------------------
```

Figure 9: Each account type has a limit for the number of books it is allowed to borrow at once. Here if the user input the id and program finds out that the account is on it's limit for the number of borrowed books, this message will be outputted informing the user about the situation



```
C:\Users\amirali\Desktop\FinalProject\main.exe
--------------------------------------------------------------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option: 3
------------------
Borrowing a book...
------------------
What's the id number: 100
What is the number of the book: 10

---------------------------------------------------
You have succefuly borrowed the book The Martian
---------------------------------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option:
```

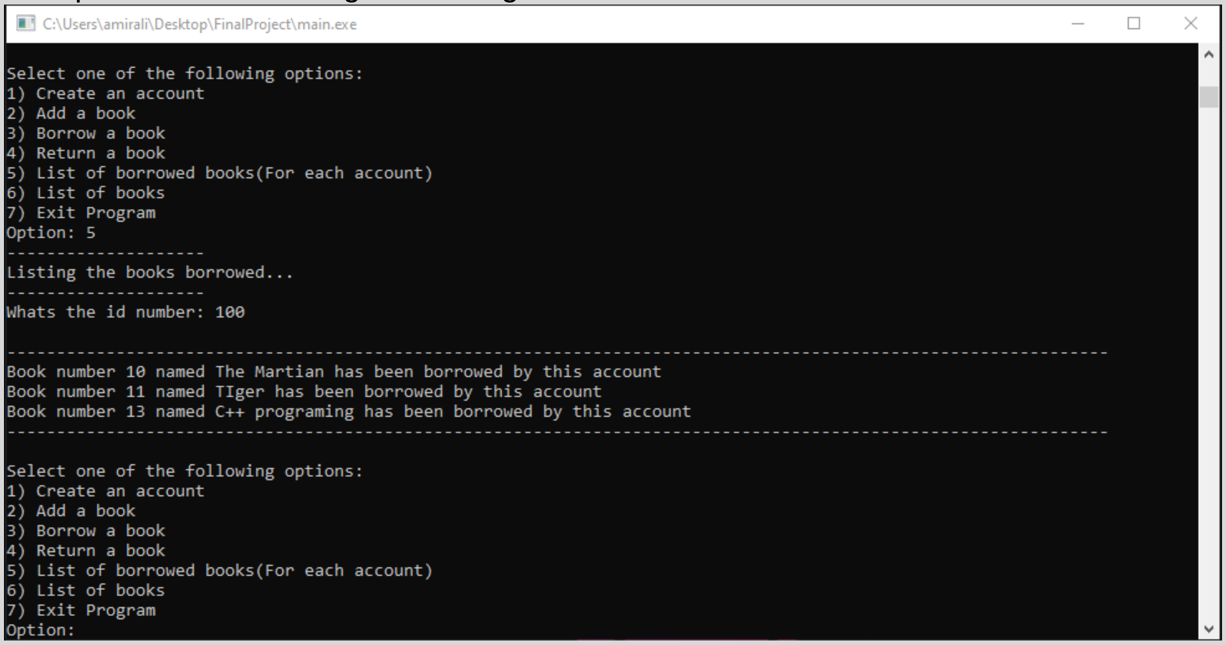
Figure 10: If user inputs the right id and book number, the book has not been borrowed and the account is not exceeding it limit for borrowed books, the operation will happen successfully and the user will borrow the book.



C:\Users\amirali\Desktop\FinalProject\main.exe

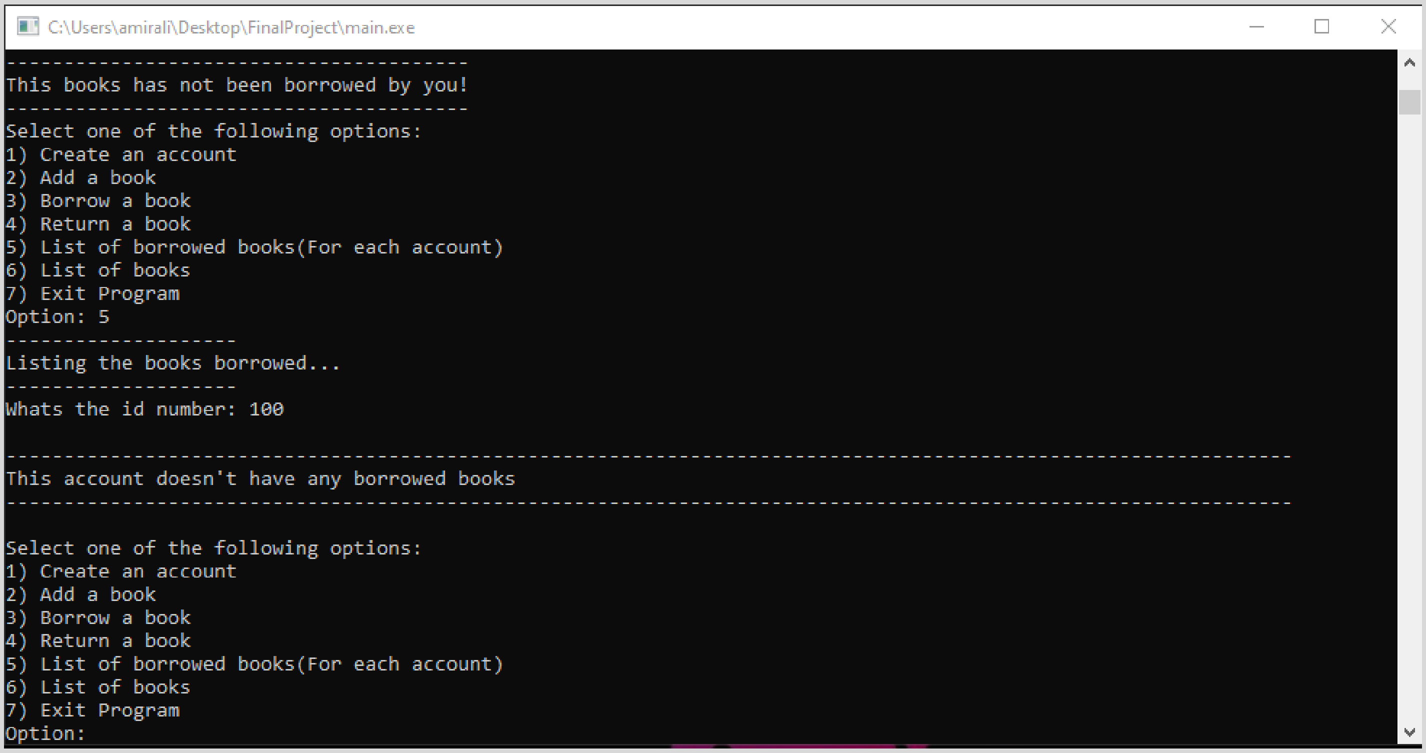
```
--------------------------------------------------------------------------
The books is already borrowed by account with the id: 100
--------------------------------------------------------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option: 4
-------------------
Returning a book...
-------------------
What's the id number: 100
What is the number of the book: 10
-------------------------------------
The book has been returned succesfuly
-------------------------------------
Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option:
```

Figure 11: Same as option 3, option 4 has id and book number validation. If both of these numbers pass and the book has been borrowed by the account associated with the inputted id number, the borrowed book will be returned successfully. If the id number doesn't match with the associated id number to the borrowed book, the program will output an error message informing the user about this situation like below:



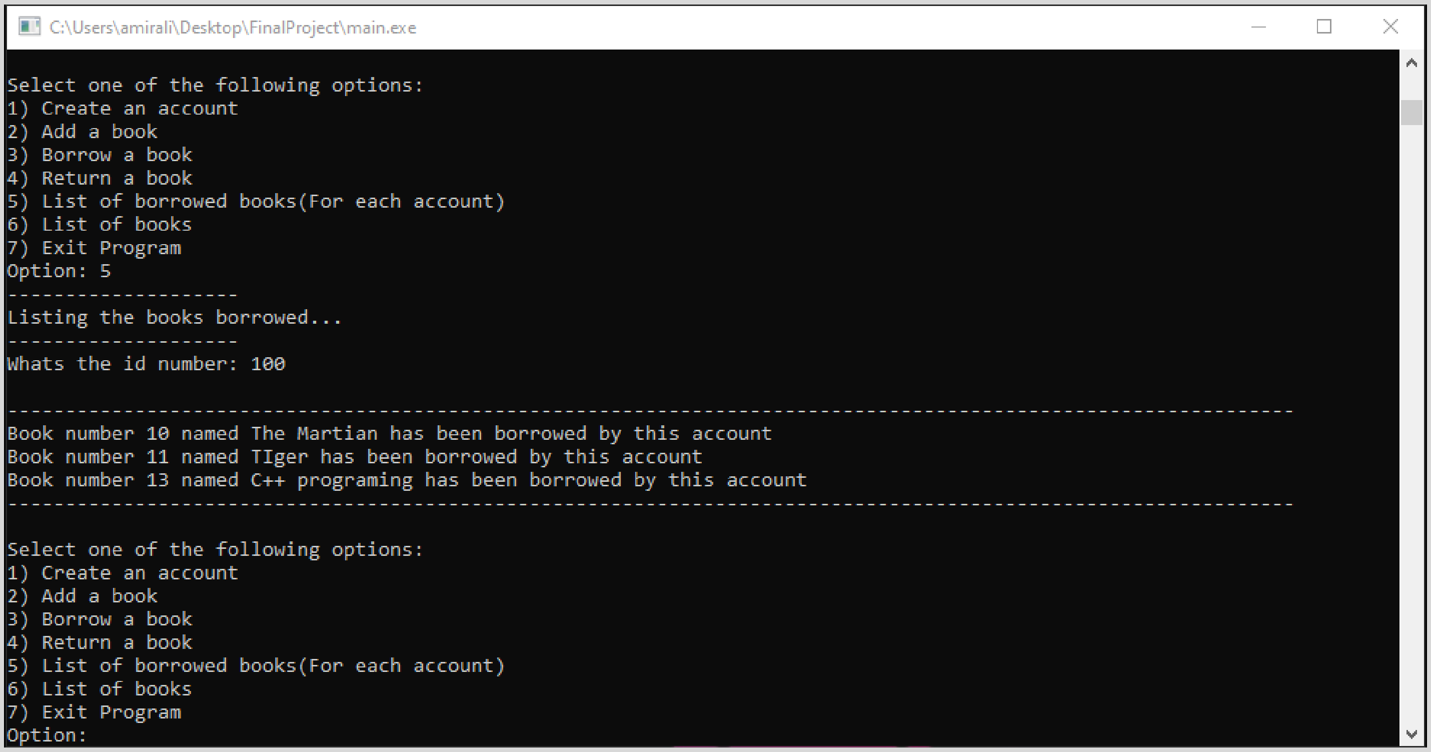
C:\Users\amirali\Desktop\FinalProject\main.exe

```

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option: 5
-------------------
Listing the books borrowed...
-------------------
Whats the id number: 100

----------------------------------------------------------------------------------------------------------------
Book number 10 named The Martian has been borrowed by this account
Book number 11 named TIger has been borrowed by this account
Book number 13 named C++ programing has been borrowed by this account
----------------------------------------------------------------------------------------------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option:
```

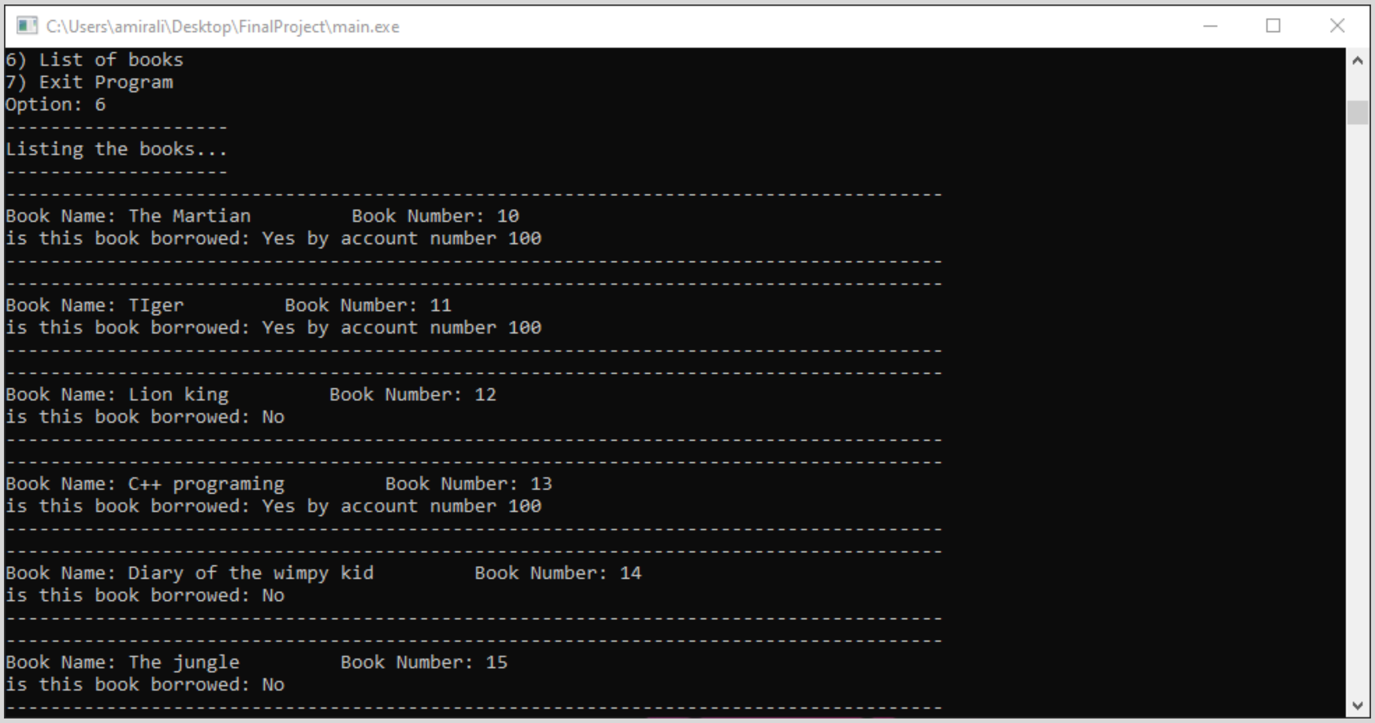


```
C:\Users\amirali\Desktop\FinalProject\main.exe
----------------------------------------
This books has not been borrowed by you!
----------------------------------------
Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option: 5
--------------------
Listing the books borrowed...
--------------------
Whats the id number: 100

----------------------------------------------------------------------------------------------------------------
This account doesn't have any borrowed books
----------------------------------------------------------------------------------------------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option:
```

Figure 12: If user selects option 5, and the entered id passes the validation process, but the account does not have any borrowed books, this message will be outputted. If the account has borrowed books the message below will be outputted:

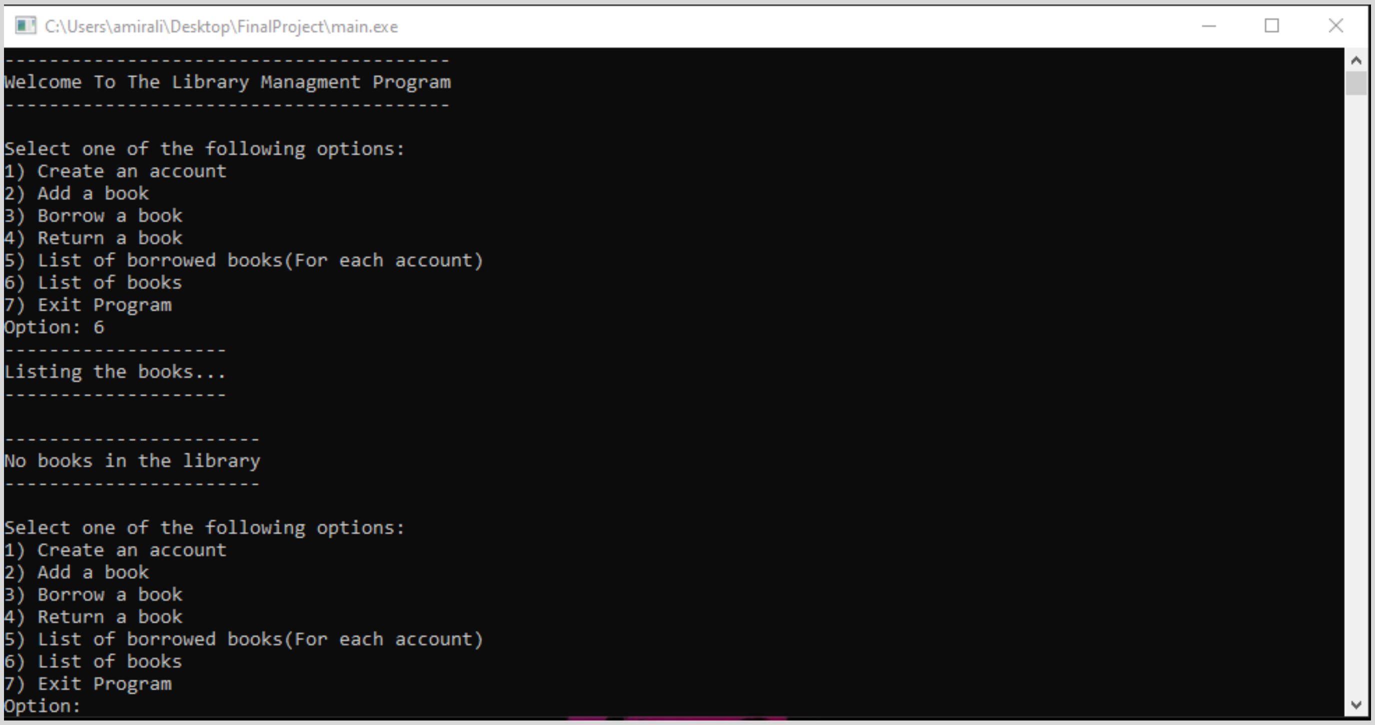


```
C:\Users\amirali\Desktop\FinalProject\main.exe

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option: 5
--------------------
Listing the books borrowed...
--------------------
Whats the id number: 100

----------------------------------------------------------------------------------------------------------------
Book number 10 named The Martian has been borrowed by this account
Book number 11 named TIger has been borrowed by this account
Book number 13 named C++ programing has been borrowed by this account
----------------------------------------------------------------------------------------------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option:
```

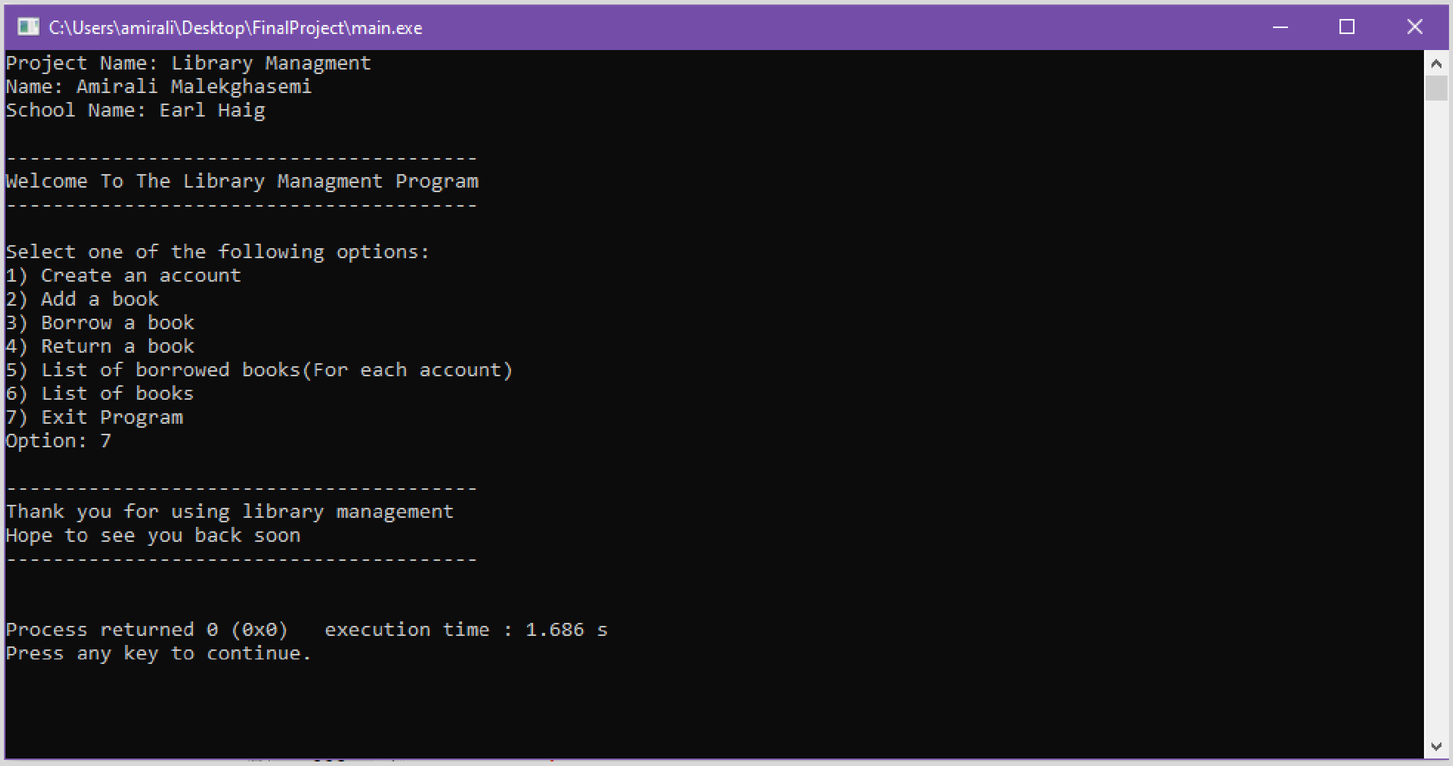


```
C:\Users\amirali\Desktop\FinalProject\main.exe
6) List of books
7) Exit Program
Option: 6
--------------------
Listing the books...
--------------------
------------------------------------------------------------------------------------
Book Name: The Martian         Book Number: 10
is this book borrowed: Yes by account number 100
------------------------------------------------------------------------------------
------------------------------------------------------------------------------------
Book Name: TIger         Book Number: 11
is this book borrowed: Yes by account number 100
------------------------------------------------------------------------------------
------------------------------------------------------------------------------------
Book Name: Lion king         Book Number: 12
is this book borrowed: No
------------------------------------------------------------------------------------
------------------------------------------------------------------------------------
Book Name: C++ programing         Book Number: 13
is this book borrowed: Yes by account number 100
------------------------------------------------------------------------------------
------------------------------------------------------------------------------------
Book Name: Diary of the wimpy kid         Book Number: 14
is this book borrowed: No
------------------------------------------------------------------------------------
------------------------------------------------------------------------------------
Book Name: The jungle         Book Number: 15
is this book borrowed: No
------------------------------------------------------------------------------------
```

Figure 13: If user selects option 6, and there are books in the library, the information about all the books will be outputted like above. If the library doesn’t have any books the message below will be outputted



```
C:\Users\amirali\Desktop\FinalProject\main.exe
----------------------------------------
Welcome To The Library Managment Program
----------------------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option: 6
--------------------
Listing the books...
--------------------

-----------------------
No books in the library
-----------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option:
```



```
Project Name: Library Managment
Name: Amirali Malekghasemi
School Name: Earl Haig

----------------------------------------
Welcome To The Library Managment Program
----------------------------------------

Select one of the following options:
1) Create an account
2) Add a book
3) Borrow a book
4) Return a book
5) List of borrowed books(For each account)
6) List of books
7) Exit Program
Option: 7

----------------------------------------
Thank you for using library management
Hope to see you back soon
----------------------------------------


Process returned 0 (0x0)   execution time : 1.686 s
Press any key to continue.
```

Figure 14: If user choses option 7 to exit the program, a warm and friendly goodbye message will be outputted

The program is coded to be as straight and user-friendly as possible with obvious error messages and lines and spaces to make the info more readable.