|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **GROUP ASSIGNMENT**    **TECHNOLOGY PARK MALAYSIA**    **CT077-3-2-DSTR**    **DATA STRUCTURES**  **GROUP ASSIGNMENT**   |  |  | | --- | --- | | Name | Tp Number | | Md Nowshad Ul Alam | Tp057903 | | Khadija Fyruz | Tp058379 | |

Table of Contents

[1.0 Introduction 6](#_Toc79335824)

[2.0 Proposed data structure: 6](#_Toc79335825)

[2.1 Description 7](#_Toc79335826)

[2.2 Justification 8](#_Toc79335827)

[2.3 Visualization 9](#_Toc79335828)

[3.0 Implementation 16](#_Toc79335829)

[3.1 Add Book Flowchart 16](#_Toc79335830)

[3.2 Snippet for Adding Books 17](#_Toc79335831)

[3.3 Display Book Flowchart 18](#_Toc79335832)

[3.4 Snippet for Displaying Books 19](#_Toc79335833)

[3.5 Display All Available Book Flowchart 20](#_Toc79335834)

[3.6 Snippet for Viewing available Title 20](#_Toc79335835)

[3.7 Search Book Flowchart 21](#_Toc79335836)

[3.8 Snippet for Searching Books 22](#_Toc79335837)

[3.9 Update Book Flowchart 23](#_Toc79335838)

[3.10 Snippet for Updating Books information 24](#_Toc79335839)

[3.11 Register Patron Flowchart: 25](#_Toc79335840)

[3.12 Snippet for patron Registration 26](#_Toc79335841)

[3.13 Search Patron Flowchart 28](#_Toc79335842)

[3.14 Snippet for patron Search 28](#_Toc79335843)

[3.15 Show All Patrons Flowchart with Active book borrowed: 29](#_Toc79335844)

[3.16 Snippet for patron with Active book borrowed 30](#_Toc79335845)

[3.17 Show the last 10 books borrowed by a patron: 31](#_Toc79335846)

[3.18 Snippet the last 10 books borrowed by patron 32](#_Toc79335847)

[3.19 Update Patron Info Flowchart: 32](#_Toc79335848)

[3.20 Snippet for Updating Patron Information 33](#_Toc79335849)

[3.21 Request Extension 35](#_Toc79335850)

[3.22 Snippet for request Extension 36](#_Toc79335851)

[3.23 Borrow book 37](#_Toc79335852)

[3.24 Snippet for Borrow book 38](#_Toc79335853)

[3.25 Return books 40](#_Toc79335854)

[3.26 Snippet for return book 41](#_Toc79335855)

[3.27 Delete patron 42](#_Toc79335856)

[3.28 Snippet for delete patron 42](#_Toc79335857)

[4.0 Snippet of the working program 43](#_Toc79335858)

[4.1 Main Manu Console 43](#_Toc79335859)

[4.2 Registration Patron 43](#_Toc79335860)

[4.3 View Personal Details 44](#_Toc79335861)

[4.4 Add Books 44](#_Toc79335862)

[4.5 Borrow Books 45](#_Toc79335863)

[4.6 Search patron 46](#_Toc79335864)

[4.7 Updating personal Information 46](#_Toc79335865)

[4.8 Patron with Active Borrowed Books 47](#_Toc79335866)

[4.9 View last 10 borrowed books 47](#_Toc79335867)

[4.10 Return books 48](#_Toc79335868)

[4.11 Request Extension 48](#_Toc79335869)

[4.12 Display all books 49](#_Toc79335870)

[4.13 Display available books 50](#_Toc79335871)

[4.14 Search Books 50](#_Toc79335872)

[5.0 Limitation 51](#_Toc79335873)

[6.0 Matrix 51](#_Toc79335874)

[7.0 References 52](#_Toc79335875)

**List of Figure**

[Figure 1: Memory Location 6](#_Toc79335876)

[Figure 2:Doubly Linked List 7](#_Toc79335877)

[Figure 3:Doubly Linked List (Borrow Books) 9](#_Toc79335878)

[Figure 4: Doubly Linked List (Return Books) 9](#_Toc79335879)

[Figure 5: Doubly Linked List (Request Extension) 10](#_Toc79335880)

[Figure 6:Doubly Linked List (Search Book) 10](#_Toc79335881)

[Figure 7: Doubly Linked List (Add Books) 11](#_Toc79335882)

[Figure 8: Doubly Linked List (Update Book Information) 11](#_Toc79335883)

[Figure 9:Doubly Linked List (Register Patron) 12](#_Toc79335884)

[Figure 10:Doubly Linked List (Borrowed Book Status) 13](#_Toc79335885)

[Figure 11:Doubly Linked List (Last 10 Book Borrowed Patrons) 14](#_Toc79335886)

[Figure 12:Doubly Linked List (Update Patron Information) 14](#_Toc79335887)

[Figure 13: Add Book Flowchart 15](#_Toc79335888)

[Figure 14: Option To Add Book code Snippet 16](#_Toc79335889)

[Figure 15:Storing Book Code Snippet 17](#_Toc79335890)

[Figure 16: Adding Book code Snippet 17](#_Toc79335891)

[Figure 17: Display All Books Flowchart 18](#_Toc79335892)

[Figure 18: Display All Books Code Snippet 19](#_Toc79335893)

[Figure 19: Display All Available Books Flowchart 19](#_Toc79335894)

[Figure 20: Display All Available Books code Snippet 20](#_Toc79335895)

[Figure 21: Search Book Flowchart 20](#_Toc79335896)

[Figure 22: Getting Book List Size Code Snippet 21](#_Toc79335897)

[Figure 23: Searching Books Code Snippet 22](#_Toc79335898)

[Figure 24: Updating Books Info Snippet 23](#_Toc79335899)

[Figure 25: Updating Book Info Code Snippet 24](#_Toc79335900)

[Figure 26: Register Patron Flowchart 25](#_Toc79335901)

[Figure 27: Register Patron code Snippet 26](#_Toc79335902)

[Figure 28: Adding Patron In the List 27](#_Toc79335903)

[Figure 29: Search Patron Flowchart 28](#_Toc79335904)

[Figure 30: Search Patron Code Snippet 28](#_Toc79335905)

[Figure 31: Patron with Active Books Flowchart 29](#_Toc79335906)

[Figure 32: Display patron with Active Books Code Snippet 30](#_Toc79335907)

[Figure 33: Display Last 10 borrowed Books 31](#_Toc79335908)

[Figure 34: Last 10 Borrowed Books By Patron 32](#_Toc79335909)

[Figure 35: Update patrons Information Flowchart 33](#_Toc79335910)

[Figure 36: Updating Patron Information option code Snippet 34](#_Toc79335911)

[Figure 37: Updating Patron Information code Snippet 34](#_Toc79335912)

[Figure 38: Request Extension Flowchart 35](#_Toc79335913)

[Figure 39: Display Current Information code Snippet 36](#_Toc79335914)

[Figure 40: Adding New Date Code Snippet 36](#_Toc79335915)

[Figure 41: Borrow Books Flowchart 37](#_Toc79335916)

[Figure 42: Showing Option to Borrow Book Code Snippet 38](#_Toc79335917)

[Figure 43: Borrow Books Code Snippet 39](#_Toc79335918)

[Figure 44: Return Books Flowchart 40](#_Toc79335919)

[Figure 45: Return Books Code Snippet 41](#_Toc79335920)

[Figure 46: Delete Patron Flowchart 42](#_Toc79335921)

[Figure 47: Delete Patron Code Snippet 42](#_Toc79335922)

[Figure 48: Main Menu Console 43](#_Toc79335923)

[Figure 49: Registration Patron 43](#_Toc79335924)

[Figure 50:View Personal Details 44](#_Toc79335925)

[Figure 51: Option to Add Books 44](#_Toc79335926)

[Figure 52: Adding Books 45](#_Toc79335927)

[Figure 53: Borrow Books 45](#_Toc79335928)

[Figure 54: Search Patron 46](#_Toc79335929)

[Figure 55:Update Personal Information 46](#_Toc79335930)

[Figure 56: Patron with Active Books 47](#_Toc79335931)

[Figure 57: View Last 10 Book Borrowed by Patron 47](#_Toc79335932)

[Figure 58: Current Books Information 48](#_Toc79335933)

[Figure 59: Returning the Books 48](#_Toc79335934)

[Figure 60: Information of Books After Returning 48](#_Toc79335935)

[Figure 61: Current Information of Patron 49](#_Toc79335936)

[Figure 62: New Date for Extension 49](#_Toc79335937)

[Figure 63: Information of Patron After getting Extension 49](#_Toc79335938)

[Figure 64: Display all List of the Books 49](#_Toc79335939)

[Figure 65: Display Available Title 50](#_Toc79335940)

[Figure 66: Search Books 50](#_Toc79335941)

# 1.0 Introduction

This assignment mainly focuses on developing a new management system based on console-based application where the main goal is goal is to manage books.

Simply Data Structure can be inscribed as a defined class of data elements which solves data storage and organizing problems and provides apt ways to use the data elements skillfully in future. (Point, 2021)

There is a simple way to define data structure in conclusion. That is:

* Programs= Algorithms + Data Structure
* Data Structures= Related data + Allowed operations on that data. (DEVMOUNTAIN, 2021)

The system allows the admins to add books, manage inventory and many other features. Users will also be able to view clearly of each type as well as the prices. They also can see the purchase history and do purchases or borrow. People who will delay returning the borrowed book will also be charged with fines. This prototype was developed in code blocks.

# 2.0 Proposed data structure:

**Data Structure** is the process through which data can be sorted out for effective use in future. Such as, by using array data structure, a group of same data-type items can be stored. (GeeksforGeeks, 2021)

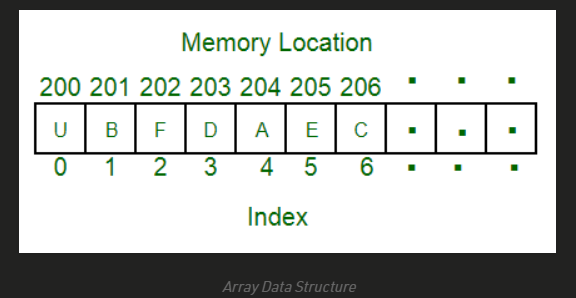


Figure : Memory Location

Data Structure is important to design algorithms effectively, as well as allowing the information to be safe while storing. Using it we can also store data on cloud which can be easily accessed from anywhere using any machine which is connected to internet. Also, a blessing in disguise is that, except experts or admins nobody can make any change in the system, so it remains safe from cracking illegally. (TANNER, 2021)

In this console-based application we will be using doubly linked list instead of singly so that we can have less time complexity and by following doubly linked list the maximum time complexity we will have in our code is log(n) and minimum will be log (1) and in some case it will be log(n/2) if we search by dividing middle node. By using doubly linked list every time we enter new records at the end the time complexity will be log (1) where in singly it is log(n) again when we need to show the last 10 records by utilizing doubly linked list our time will be log(n) whereas in singly it will be log(n^2). Again, in the same way if we use array, we will have space complexity while we will be going to resize our array. So, for this console-based application doubly linked list would a better choice in case of both time and space complexity.

## 2.1 Description

In this assignment, our group decided to propose **Doubly Linked List** for the community library application. The navigation of doubly linked list is possible in both ways, both forward and backward and as it is easier to navigate compared to singly linked list, is the cause why it is called doubly linked list.

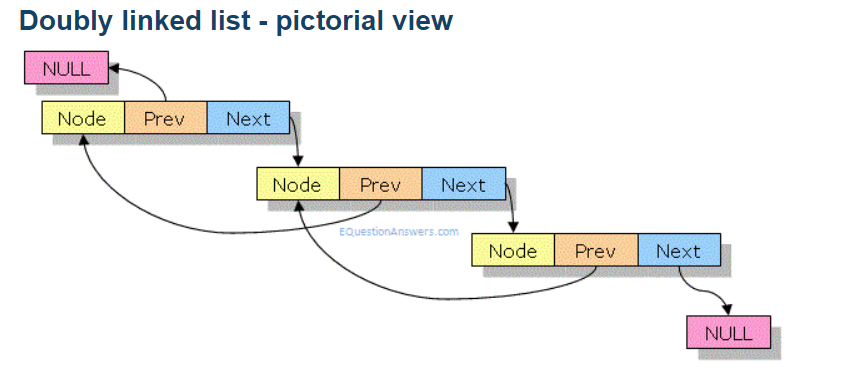


Figure :Doubly Linked List

These are the terms for understanding the Doubly Linked List concept efficiently: (TUTORIALSPOINT, 2021)

1. **Link –** Each link of a linked list stores data which is called element,
2. **Next –** A link to the next link is contained by each link which is the Next,
3. **Prev -** A link to the previous link is contained by each link which is the Prev,
4. **LinkedList –** It contains the connection link to the first and last link. The connection to the first link is called First and the last link is called Last. (TUTORIALSPOINT, 2021)

In this Community library management system, we will use doubly linked list to manage:

* Book data including book title, writer, genre and other detailed info for book storage.
* Book storage which is responsible for storing books by the admins.
* Existing book list which allows books that are available (not borrowed) and can be allotted inside it.

The column nodes will consist of two pointers which are the “Book\*next” and “Book\*prev” which will store the address of the next column node and previous column node.

## 2.2 Justification

A doubly linked list (DLL) contains an extra pointer than singly linked list (SLL) which is called the Previous Pointer, together with the data and Next Pointer. (GeeksforGeeks, 2021)

There are some advantages for which our team has chosen the Doubly linked list to develop this system.

* It can be easily moved forward and backward.
* In the doubly linked list, the “Delete” option is more efficient if the node pointer to be deleted is given.
* In this DLL, to add a new node before a given node is easier and quicker.
* One advantage compared to the singly linked list is, in the singly linked list, to get the previous node, the whole list is traversed sometimes. But in the doubly linked list it can easily be found using the “Prev” pointer. (GeeksforGeeks, 2021)

## 2.3 Visualization

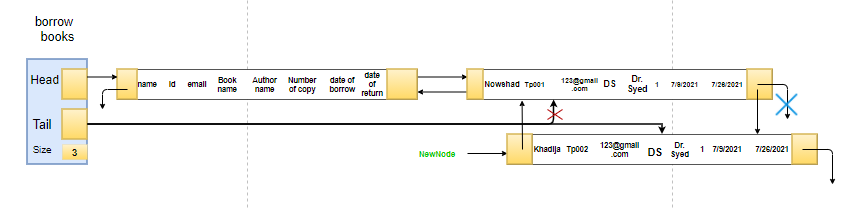


Figure :Doubly Linked List (Borrow Books)

The above visualization represents how we are going to implement borrow books function here if a patrons borrow books from library, it will create a new node in patron borrowing books list and the books copy will be reduced from the book list. As instance we have 4 copies of a book which name is data structure and if a patron takes one copy from that book in book list the copy will be 3 and a new node will create where it will store all information of the patron including number of copy and book and author name. here to avoid time complexity we are going to use tail as our start node while we are going to create new node of the patron in active borrowed book status and in that way time complexity will be log (1).

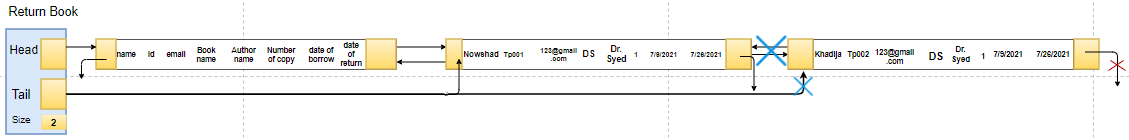


Figure : Doubly Linked List (Return Books)

In this function our patron will return book by using their name and if the name match with the active patron they can return their book and their active status will be deleted if their information including book name and his/her name. here time complexity will be log(n).

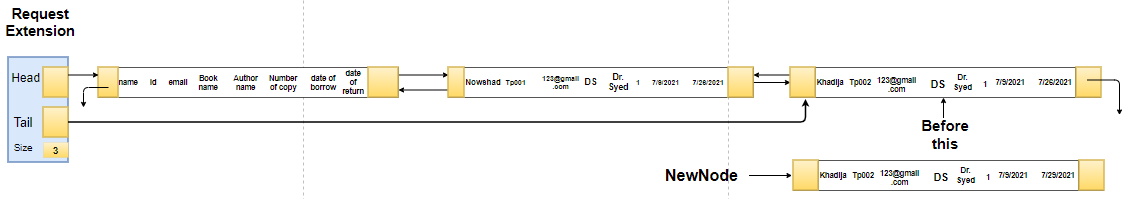


Figure : Doubly Linked List (Request Extension)

In this visualization , it represts the function of the doubly link list that it will be going to do extension of the date. Here this extension is similar to updating function but here we can only update date and the time complexity will be log(n).

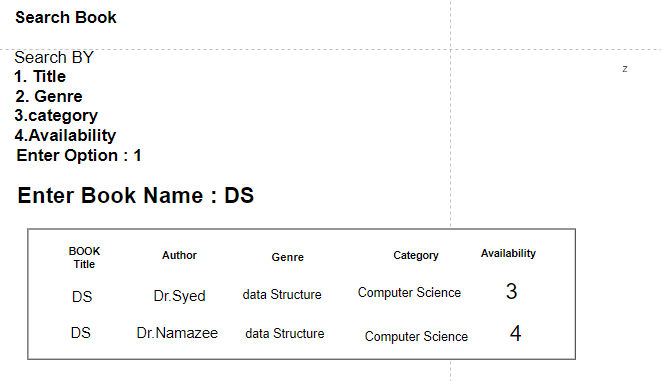


Figure :Doubly Linked List (Search Book)

This is the visualiztion of how we are going to search book, here we can search book by title , genre, category, availability. If they information above match than we can view all option stored in the list including copy,name of book, author, category etc. the time complixity of this part will be log(n).

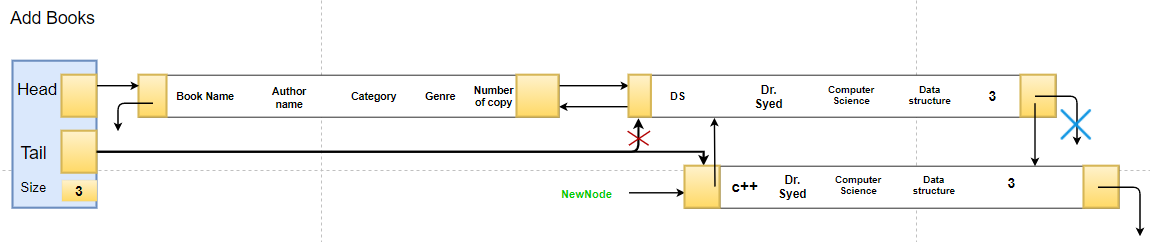


Figure : Doubly Linked List (Add Books)

In this part we will be using tail as our start point to avoid time complexity and if the use tail as start points the time complexity will be log (1) as will stored every new node at the end.

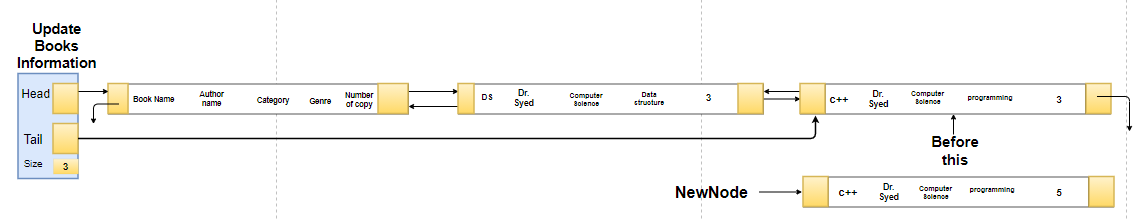


Figure : Doubly Linked List (Update Book Information)

This part is where we will be updating book information, we can update any information of the book including number of copy and books name and in this part the time complexity will be log(n).

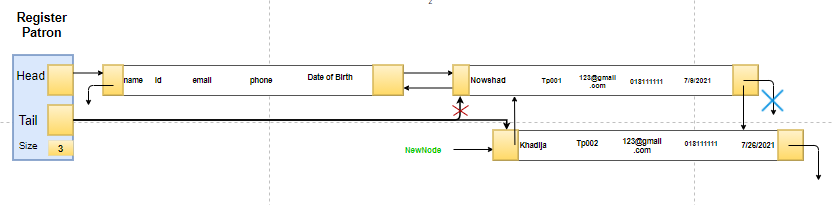


Figure :Doubly Linked List (Register Patron)

In this part, we will be registering new patron by their name email phone and with date of birth after doing registration an automatic id will generate for this particular patron name which will be unique. Here we will be using doubly linked list because if we want to see recent register patron, we can easily see it without having much space and time complexity here if we follow doubly linked list in register patron time complexity will be log (1) instead of log(n) in singly linked list.

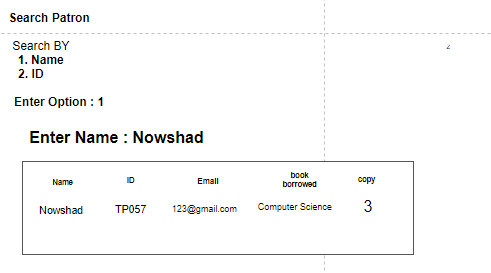


Fig 2.3.8: Doubly Linked List (Search Patron)

This is the visualiztion of how we are going to search patron,In this Library management console based application, we can search register patron by their name and unique id.

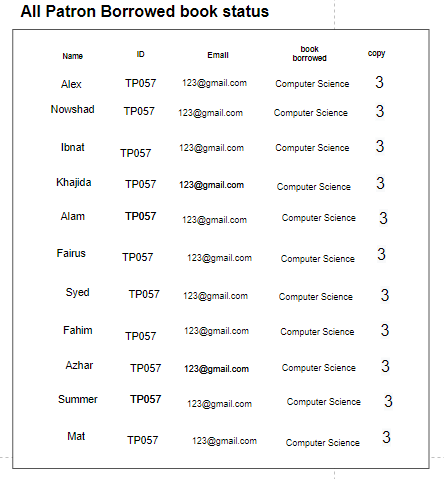


Figure :Doubly Linked List (Borrowed Book Status)

This is the part where we can search or view all the register patron with active borrowed book. We can view all their information including name, id, email, category of the book and the number of copy they have taken.

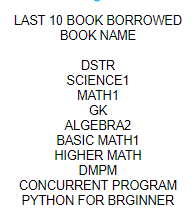


Figure :Doubly Linked List (Last 10 Book Borrowed Patrons)

This is the visualization of the part where it represents last 10 registers patron who borrower book. We will be using doubly linklist so that by using tail we can easily see last 10 patrons who borrowed book instead of using reverse function in singly linklist so that in this way we can get rid from time complexity which will be log(n) intead of log(n^2) .

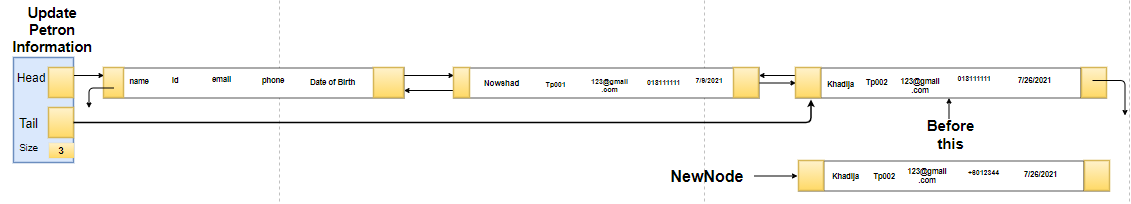


Figure :Doubly Linked List (Update Patron Information)

This is the visualization of updating information of the patron where we will be using doubly linked list as we are using all in doubly linked list so it will be better to use this here two, but the time complexity will be log(n).

# 3.0 Implementation

## 3.1 Add Book Flowchart

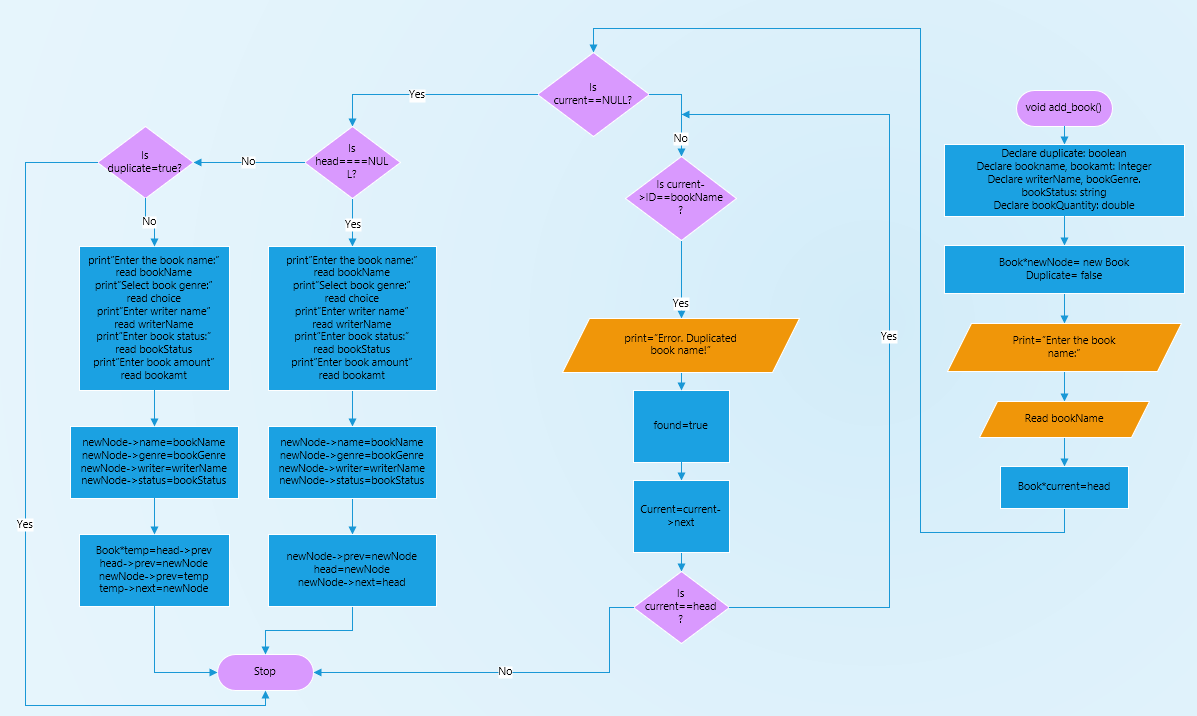


Figure 13: Add Book Flowchart

In this application while users will be going to add books, it will ask the category name after selection category they can add books what they want. Here As we mentioned before I used singly linked list as add in the front that is in the beginning will have log (1) time complexity and will have no space complexity issue if we add this in front so for having this advantage in singly linked list, I implemented add book part using singly linked list.

## 3.2 Snippet for Adding Books

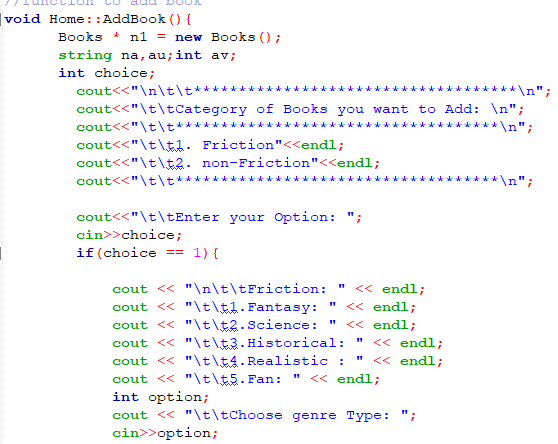


Figure 14: Option To Add Book code Snippet

Above figure 2 shows how one can add books in the library where program will first ask to select in which category and genre, they want to add books and after that they need to add the author’s name, book name and copy number which is shown in below figure 3

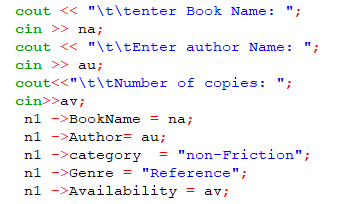


Figure 15:Storing Book Code Snippet

In the figure above there I showed only for adding book in non-friction category and in reference genre. In this way I add every condition given in program to add books.

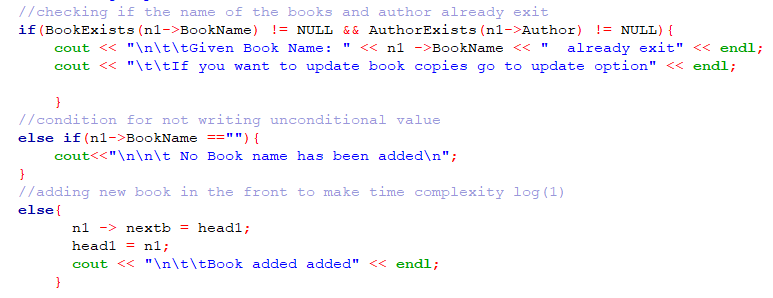


Figure 16: Adding Book code Snippet

This is the last condition in adding book’s part. Here if the same book and written by same author name already exist users cannot add this book again instead users need to update information related to that. And if the book doesn’t exist new book will be added in the library list which is shown in above figure 4.

## 3.3 Display Book Flowchart

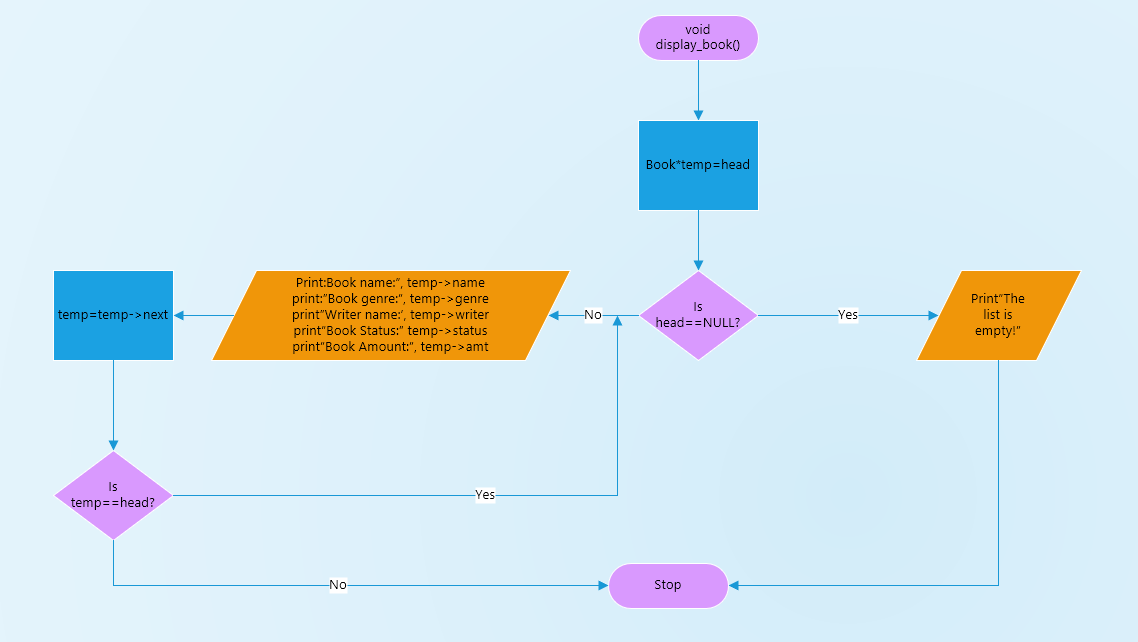


Figure 17: Display All Books Flowchart

As in printing all books we need to show all books that exits in category so we don’t need to implement doubly here because using doubly and singly linked list both with give us same space complexity and time complexity so for this part also using singly linked list would be efficient. Here I used while loop and condition until end of the last node, we can also use for loop instead of while.

## 3.4 Snippet for Displaying Books

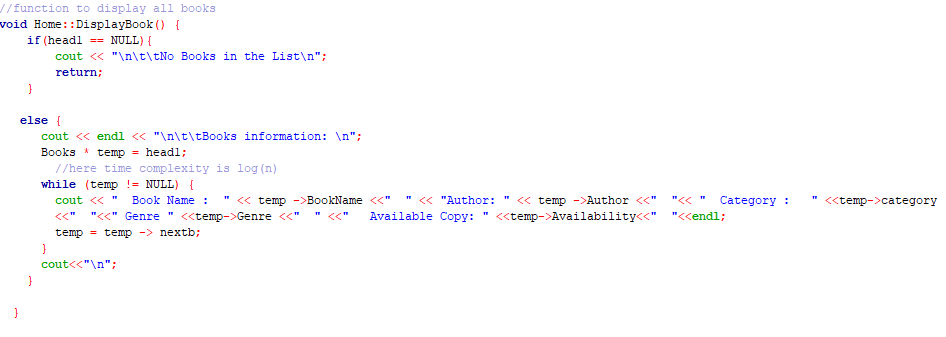


Figure 18: Display All Books Code Snippet

Above snippet shows how program will display all the books by using while loop and taking a time complexity of log(n) it will view all the books.

## 3.5 Display All Available Book Flowchart

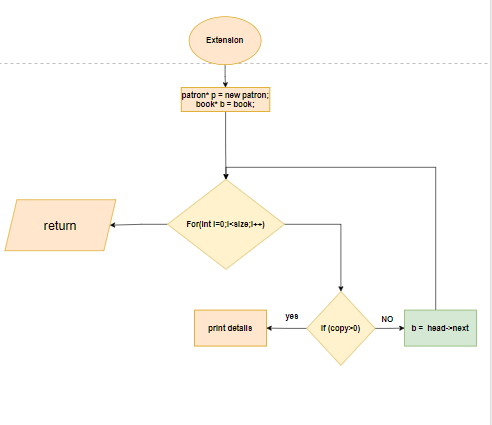


Figure 19: Display All Available Books Flowchart

The above flowchart shows how program shows available title within the library where program is implemented using for loop as it is obvious to show all the available books within the library. If we used while loop instead to for loop the program will never stops or sometimes can occurs time compilation error.

## 3.6 Snippet for Viewing available Title

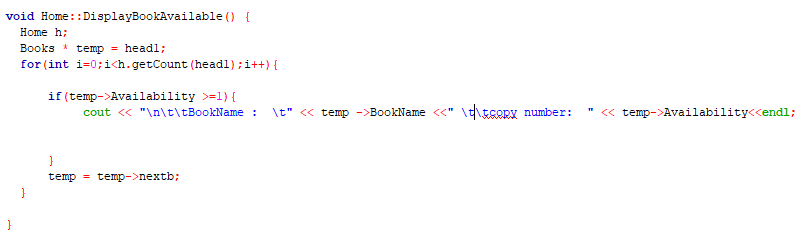


Figure 20: Display All Available Books code Snippet

## 3.7 Search Book Flowchart

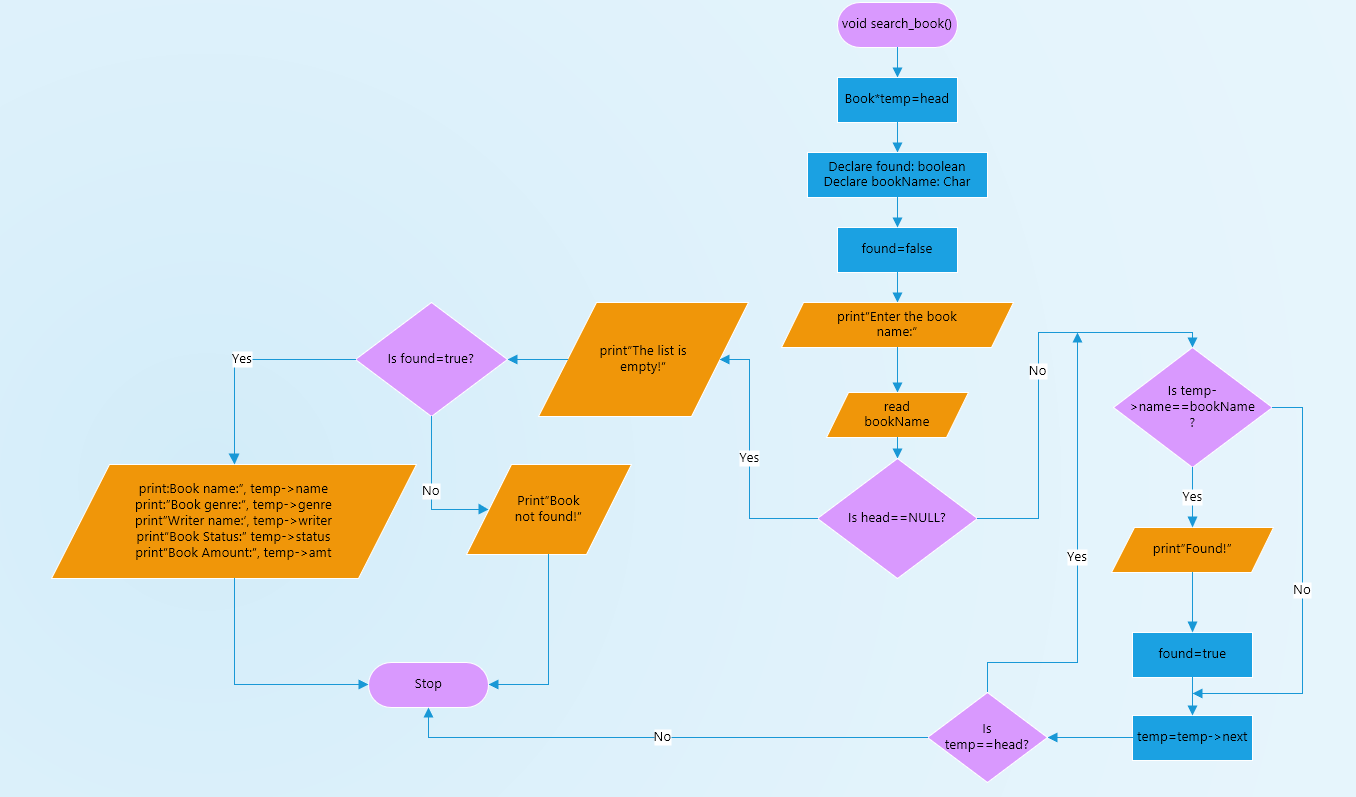


Figure 21: Search Book Flowchart

As like display function in books search also need looping for which I used linked list for this also. Here if patron wants to search for any books they can search for any option which are (category,name,genre and copy number). After that the function will have same flowchat which is shown above. If program can detect user it will stop there instantly. So time complexity will depends on the search name for instance if book stored in the beginning time complexity would be log (1), if book is in the middle time complexity would be log (n/2) and in the same way time complexity would be log(n) if book is in the end.

## 3.8 Snippet for Searching Books

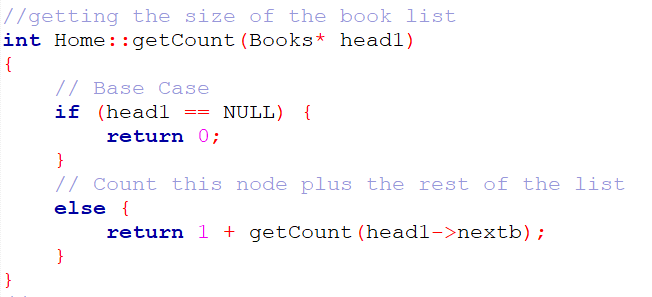


Figure 22: Getting Book List Size Code Snippet

Above figure shows how Program will count the size of the book list so that we can further used it searching books because if we search books based on category there would be many books under that category so for that we need to know the size of the list.

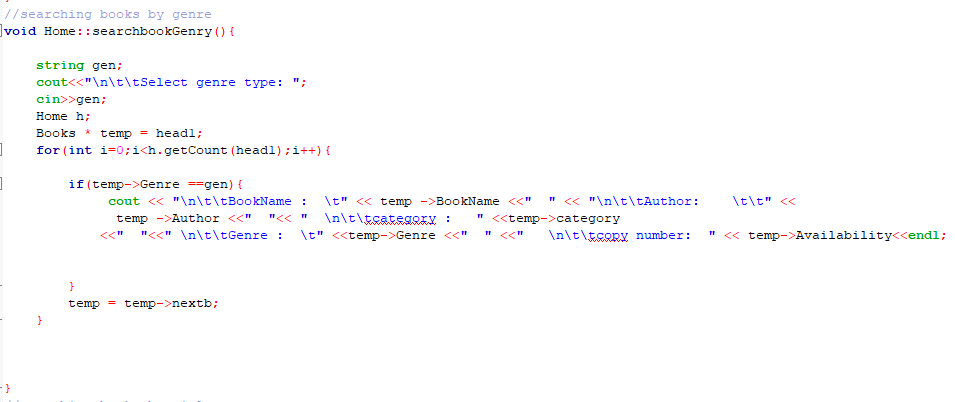


Figure 23: Searching Books Code Snippet

This figure shows how program will show the list of the books if users search for this book. However, above figure shows searching function only for genre, there are all the searching function are done following this method which are – searching by category, name, and availability.

## 3.9 Update Book Flowchart

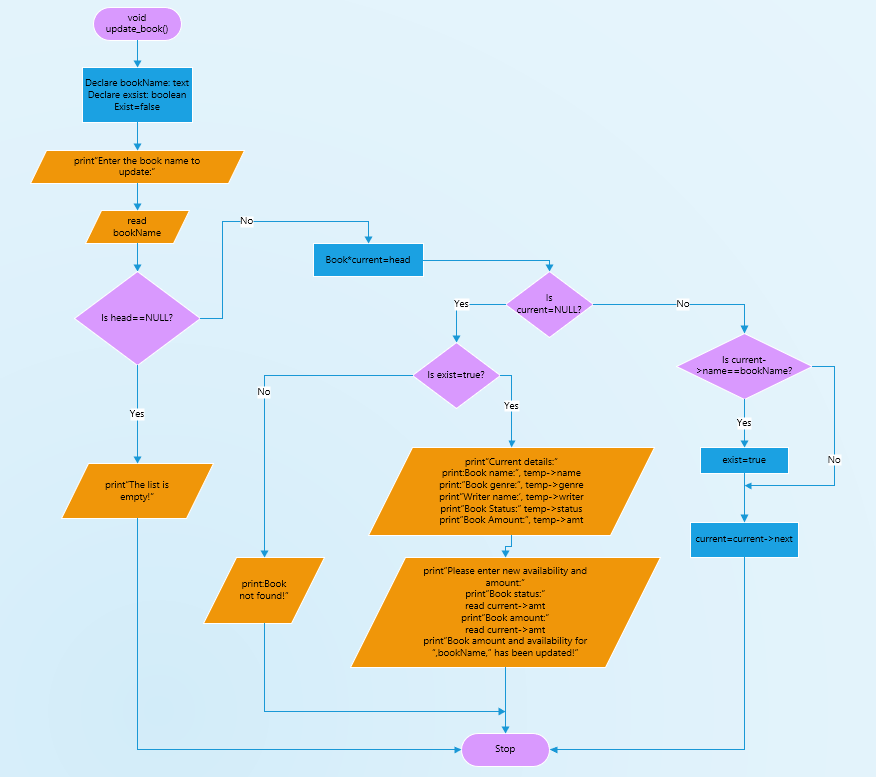


Figure 24: Updating Books Info Snippet

As updating also like search so I didn’t change data structure method here so it would same as search function which is singly linked list. Here they will have option to update them what they want. For instance, if they put wrong availability number, they will be option to update this. Time complexity would be same as same as search that is log(n).

## 3.10 Snippet for Updating Books information

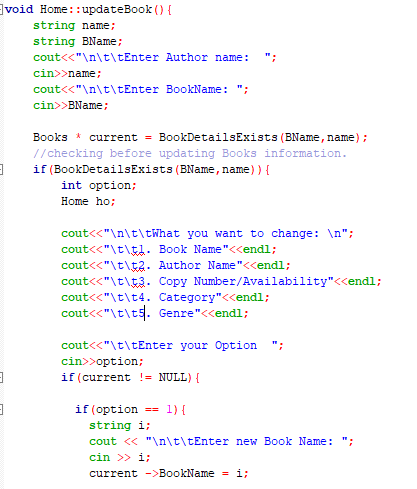


Figure 25: Updating Book Info Code Snippet

The above figure shows how program will update the information of books if users want to update any information. Here first users need to give the book name and author name if it matches with the listed books information, they will be given option to change what they want. In this figure I just showed one condition there are 4 more condition like this which are – author name, copy number, category, and genre.

## 3.11 Register Patron Flowchart:

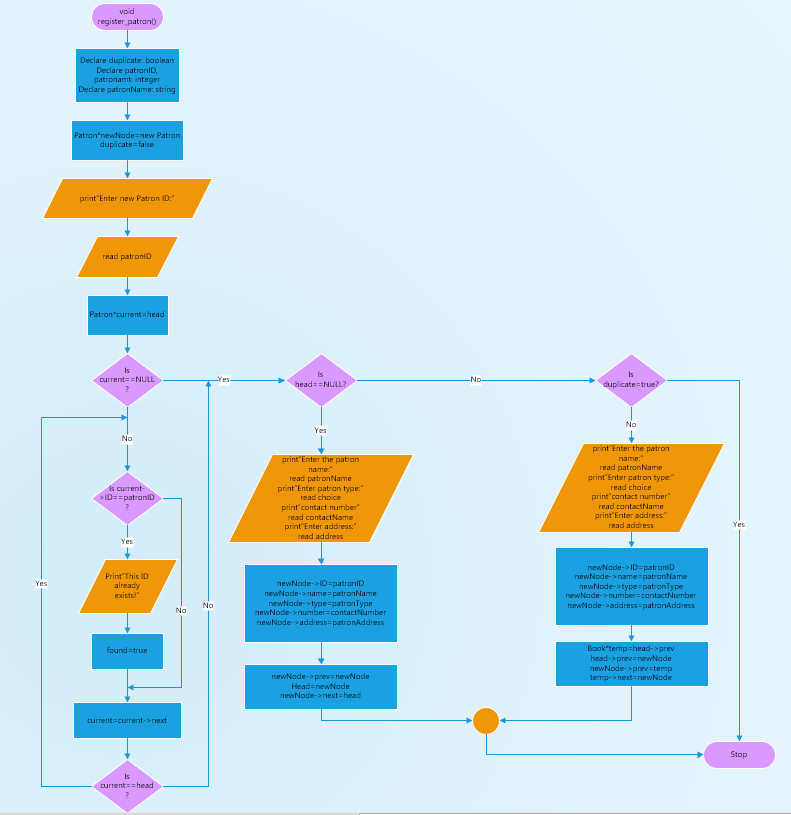


Figure 26: Register Patron Flowchart

The above figure describes the flowchart that we are going to register new patron. In the figure first we will be going to declare one new node we all the new information we be added and if the head is found null we will insert it after head but if it is not null the function will check if the id and name given by the patron already stored in the function of not if it is not found by using tail from the doubly linked list it will add the new patron at the end.

## 3.12 Snippet for patron Registration

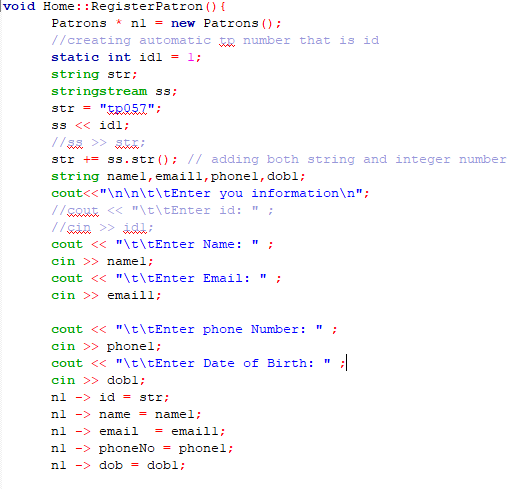


Figure 27: Register Patron code Snippet

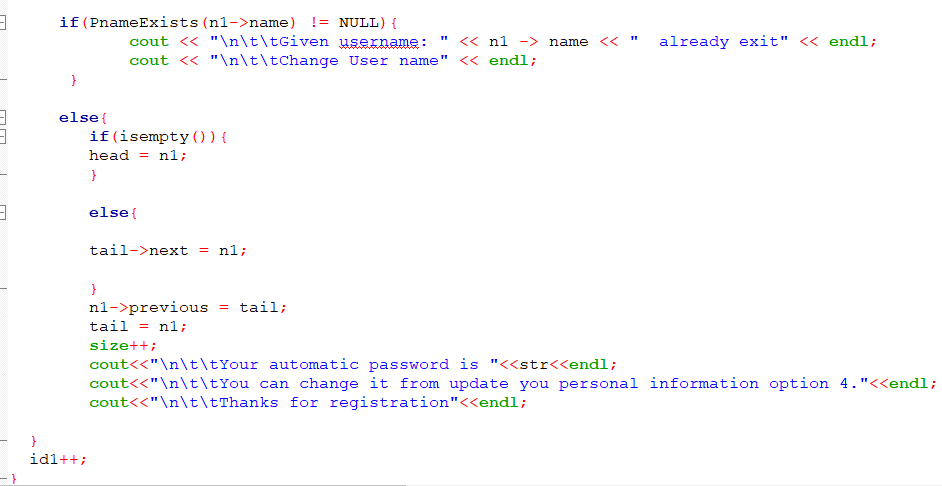


Figure 28: Adding Patron In the List

The above figure shows how patron will be registered into Library Management. To do so first they need to enter their name if name already exist, they need to change the name again and after entering all detail their name will be added to the patron list with automatic generated code which they can see after registration.

## 3.13 Search Patron Flowchart

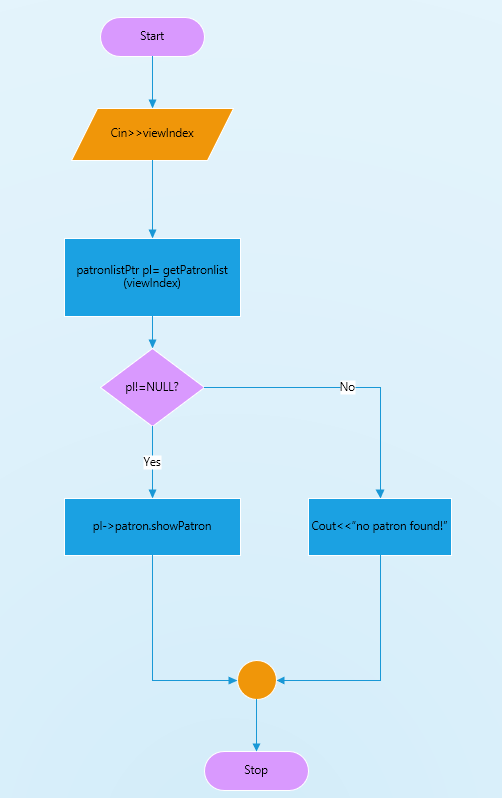


Figure 29: Search Patron Flowchart

Though we used doubly link list here, but it will still be time complexity of log(n) as to search any records program needs to go through loop one by one there will be no change in time complexity if we both singly and doubly linked list.

## 3.14 Snippet for patron Search

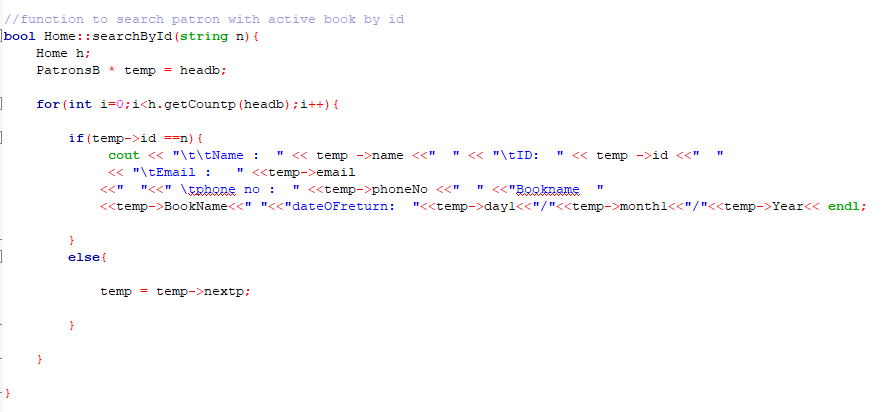


Figure 30: Search Patron Code Snippet

In the above figure, it illustrates the implementation of search function where we can see if users name match with the patron list name with active books borrowed program will show the details and this snippet shows only search with ID but in program, I did with name also. However, in the program there are two search function one is patron with active book borrowed another is patron personal information.

## 3.15 Show All Patrons Flowchart with Active book borrowed:

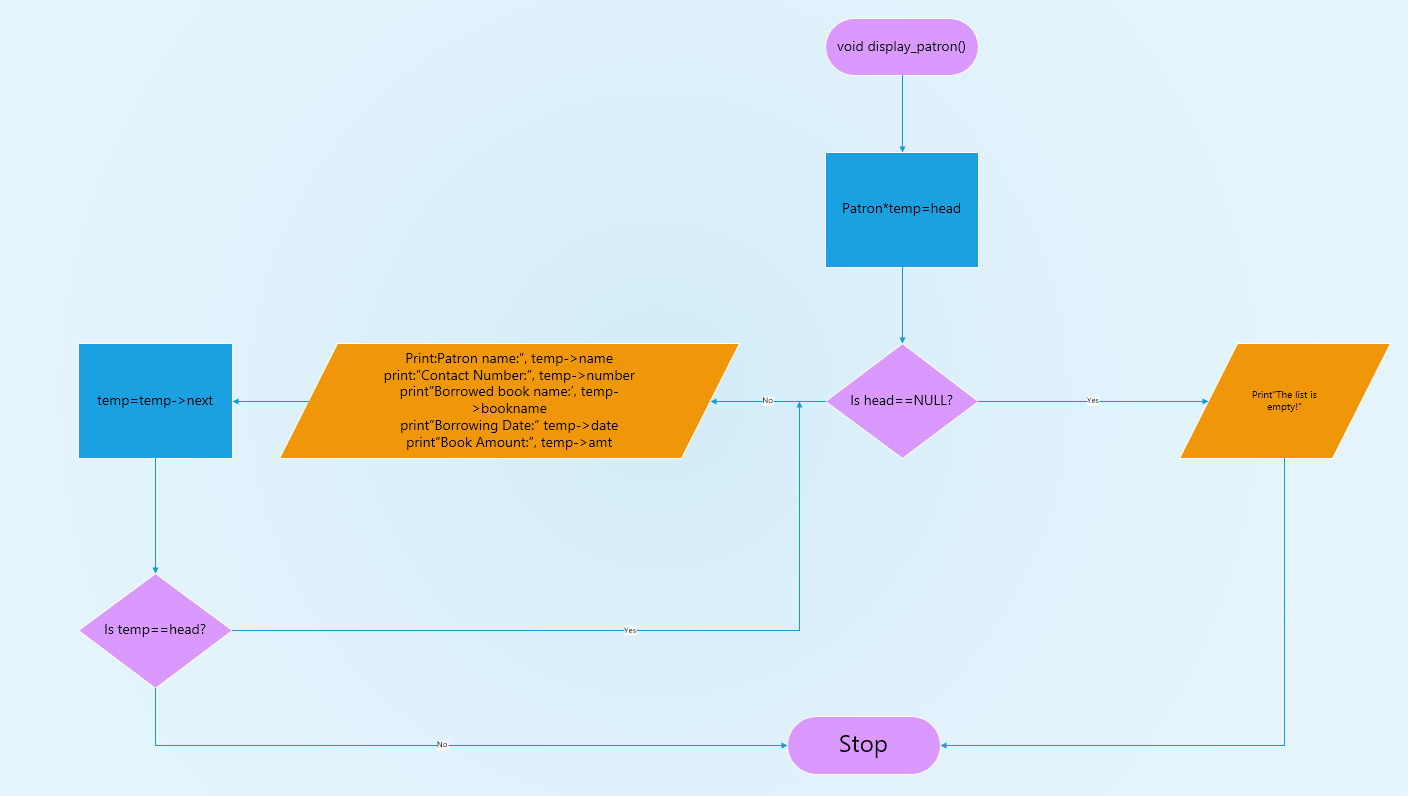


Figure 31: Patron with Active Books Flowchart

The above flowchart describe how we are going to view all the active register patron with borrowed book. Here to view all the patron with active borrowed book we will be using doubly linked list and here we will check if the patron has more than or equal to 1 copy from any types of categories if patron information match with the condition it will show us all patron otherwise it will show “No patron Found message” if there is no patron with active borrowed book and again in case of time complexity it will be log(n).

## 3.16 Snippet for patron with Active book borrowed

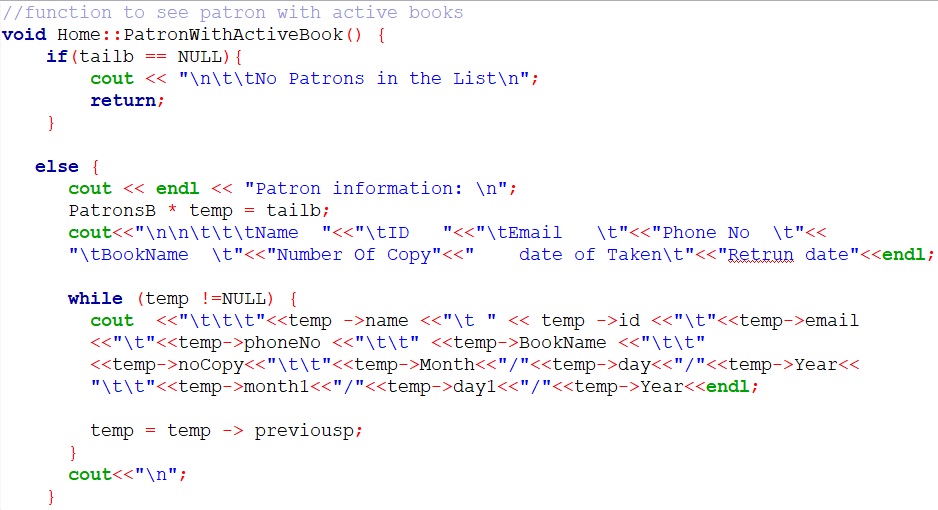


Figure 32: Display patron with Active Books Code Snippet

## 3.17 Show the last 10 books borrowed by a patron:

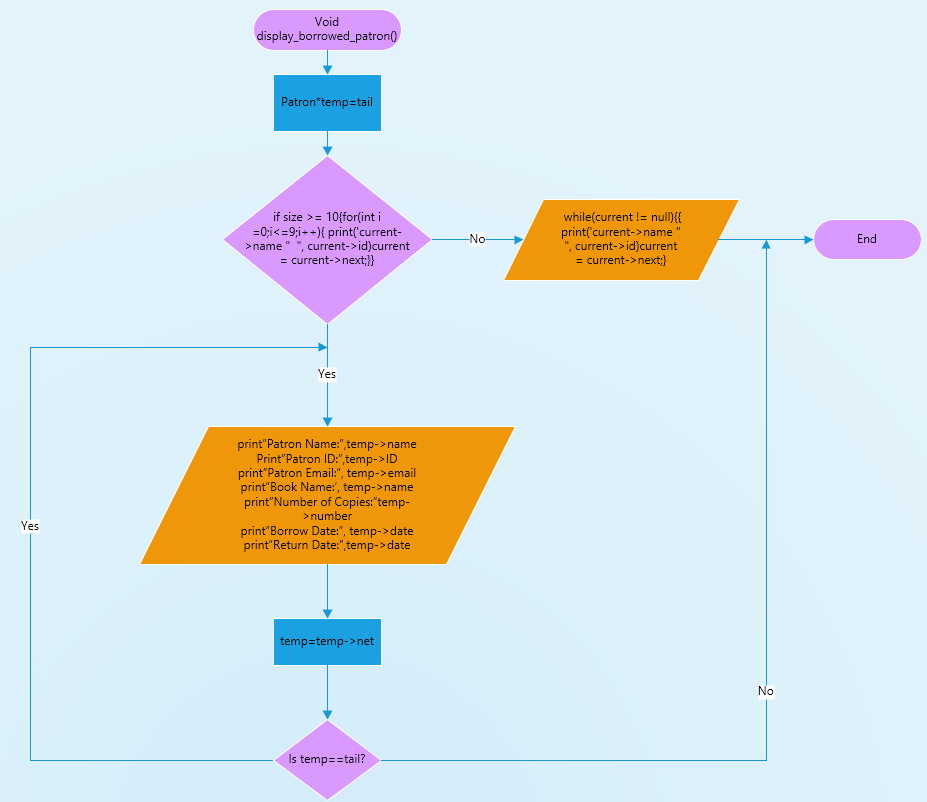


Figure 33: Display Last 10 borrowed Books

This is flowchart which represent how we are going to view last 10 patron who borrowed books and it this function we are going to use tail as our start node so that instead of going from head and doing reverse we can easily find last 10 patron with active borrowed books with a time complexity of log(n) instead of log(n^2). But again if somehow in patron records if there is less than 10 person with active borrowed books it will only show this person for example if we have only five person in our record with active borrowed books instead of showing error it will show these five which will be efficient and easy and we can run our code again and if there is not patron with active status it will show us “No active patron found” message. For this function I used a separate doubly linked list where it will take information from recent book borrowed list in FIFO method where size is 10 that means after size become 11 doubly linked list will deleted will delete one node from the front side.

## 3.18 Snippet the last 10 books borrowed by patron

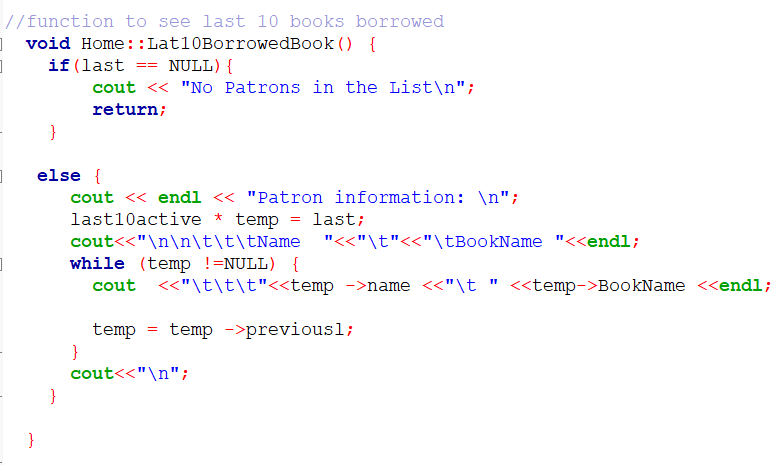


Figure 34: Last 10 Borrowed Books By Patron

## 3.19 Update Patron Info Flowchart:

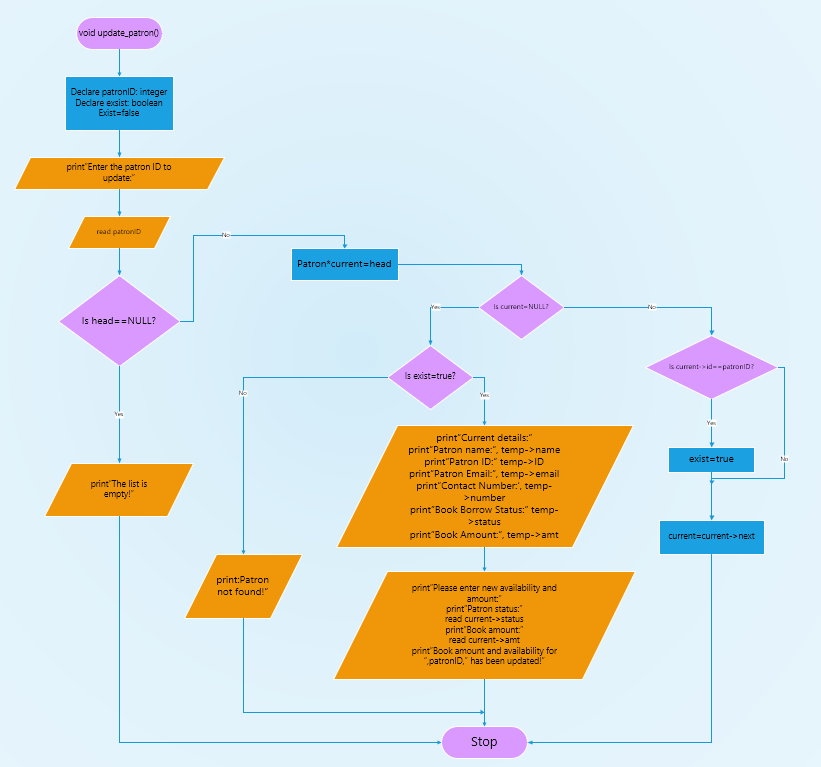


Figure 35: Update patrons Information Flowchart

In the above flowchart we can see this is the flowchart of updating patron information here as I describe in the visualization part if the enter name match with given name than the librarian will have access to change patron information like borrowed book, email, phone number etc. here again to change information or to update information time complexity will be log(n).

## 3.20 Snippet for Updating Patron Information

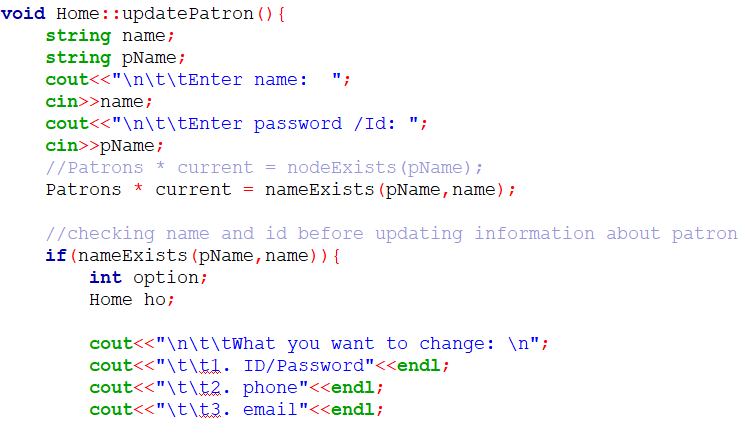


Figure 36: Updating Patron Information option code Snippet

In the update function for the patron information there will be three option to update which are id, phone number, and email. As we cannot add birthday date so it will be unchangeable. If users change any information in patron, it will also change in all lists where patron information exist.

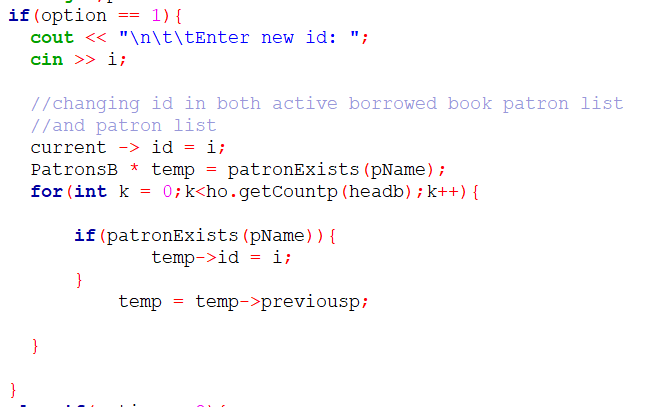


Figure 37: Updating Patron Information code Snippet

Here I showed only condition which is change ID. However, there will be 2 more condition to change the details.

## 3.21 Request Extension

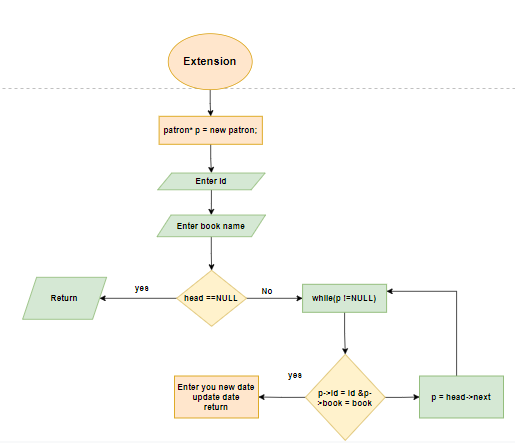


Figure 38: Request Extension Flowchart

Request extension will also act like updating information where patron can only update the return date.

## 3.22 Snippet for request Extension

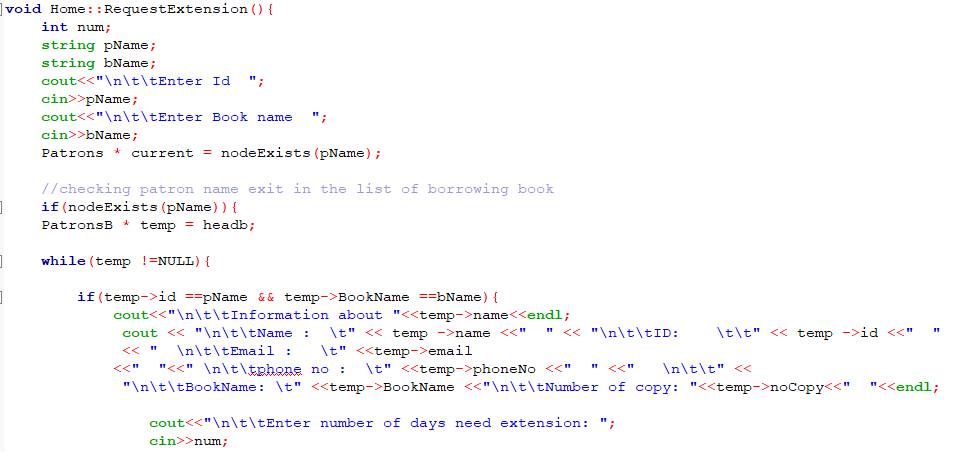


Figure 39: Display Current Information code Snippet

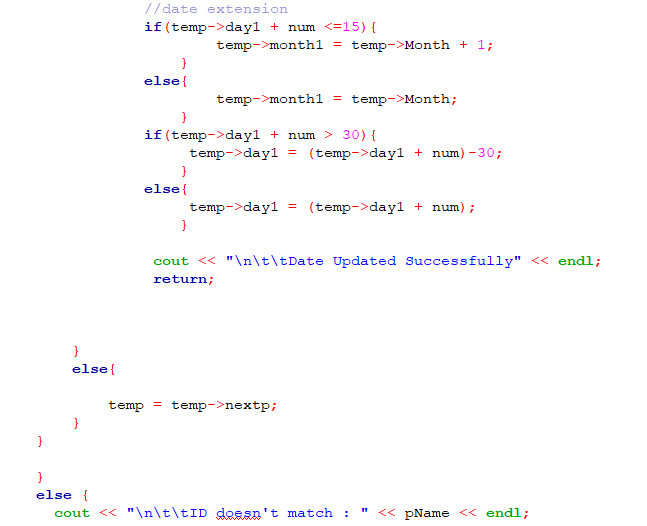


Figure 40: Adding New Date Code Snippet

Here in the above figure if users want to extend date, they need to enter name and id. If it match with the exist patron with active book, users need to add numbers of days they want and the program will automatically add it to the return date.

## 3.23 Borrow book

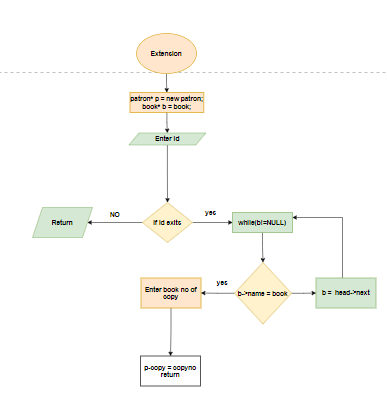


Figure 41: Borrow Books Flowchart

For this function I used separate doubly linked list as if someone return book their records will delete from active books, however it will not be deleted from the main stored data base and further.

The above visualization represents how we are going to implement borrow books function here if a patrons borrow books from library, it will create a new node in patron borrowing books list and the books copy will be reduced from the book list. As instance we have 4 copies of a book which name is data structure and if a patron takes one copy from that book in book list the copy will be 3 and a new node will create where it will store all information of the patron including number of copy and book and author name. here to avoid time complexity we are going to use tail as our start node while we are going to create new node of the patron in active borrowed book status and in that way time complexity will be log (1).

## 3.24 Snippet for Borrow book

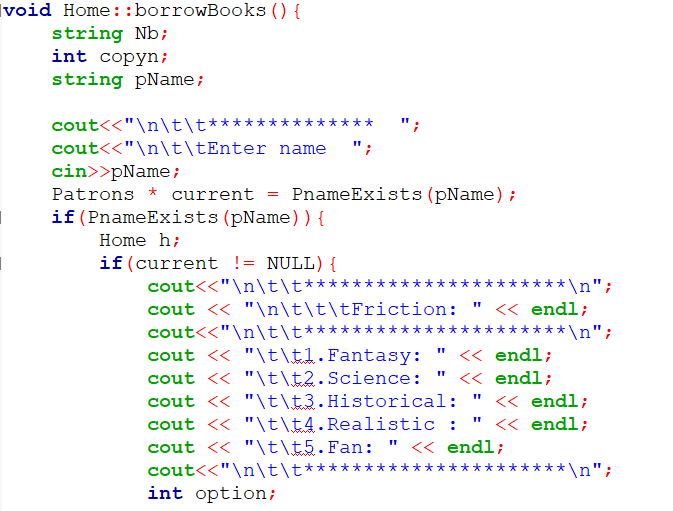


Figure 42: Showing Option to Borrow Book Code Snippet

To borrow books users first enter their name and ID, after that they will have option to borrow books on category and genre.

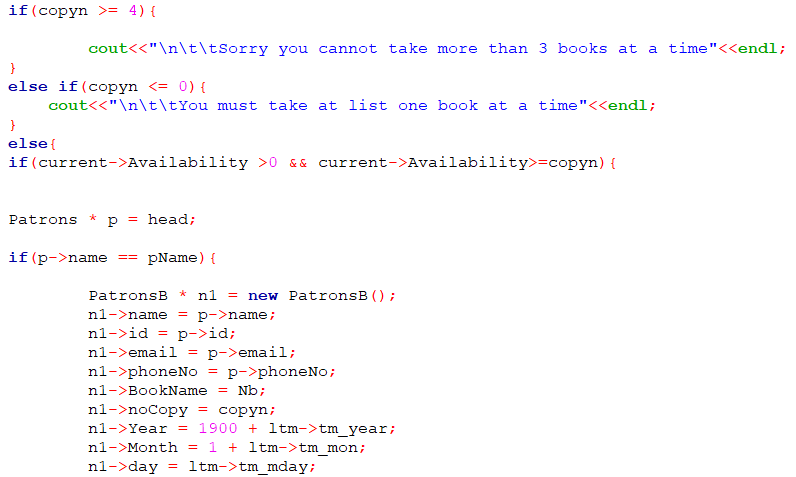


Figure 43: Borrow Books Code Snippet

If any user wants to borrow more than 3 books at a time the program will return to the main function give a message like “cannot take more than 3 books” and in the same way if number of copies is less than the number of copy users want the program again to return to the main function giving message like “Number of books not available”.

## 3.25 Return books

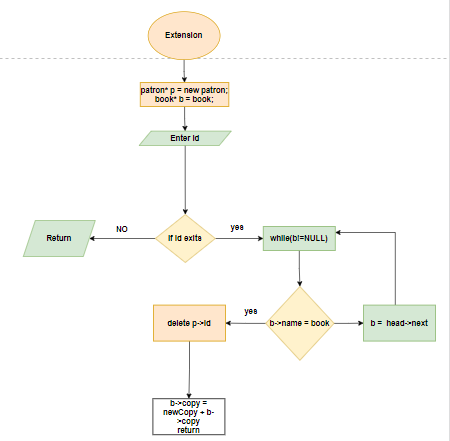


Figure 44: Return Books Flowchart

In this function our patron will return book by using their name and if the name match with the active patron they can return their book and their active status will be deleted if their information including book name and his/her name. here time complexity will be log(n). Besides that, the copy number will be added to the matched book list in the library. For example – if a person name “X” took a book from library and if the book name match in existence lists its size will reduce by one (such from 4 it will be 3) again when the same person “X” will return this book the size of the book will be again 4.

## 3.26 Snippet for return book

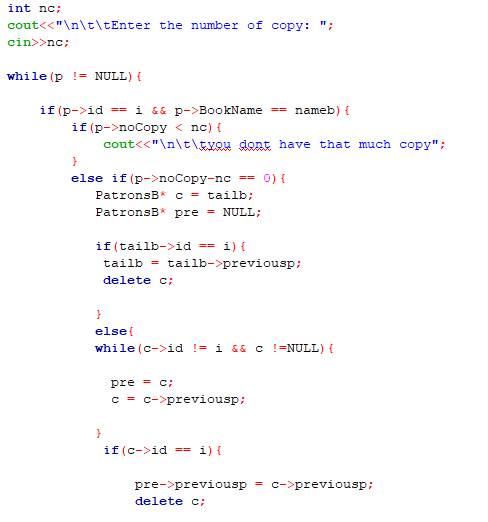


Figure 45: Return Books Code Snippet

## 3.27 Delete patron

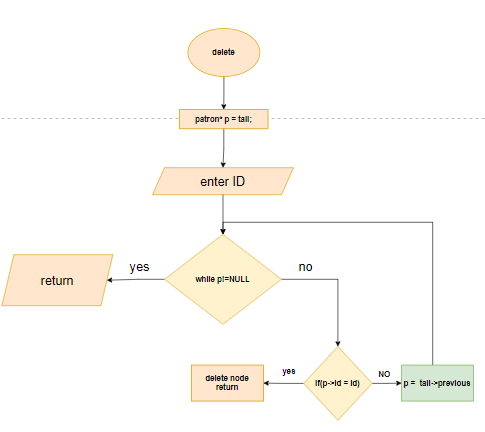


Figure 46: Delete Patron Flowchart

For deleting patron from list first users need to enter the name and id if it matches with the exist patron in the list it would delete that node which is shown in the above figure.

## 3.28 Snippet for delete patron

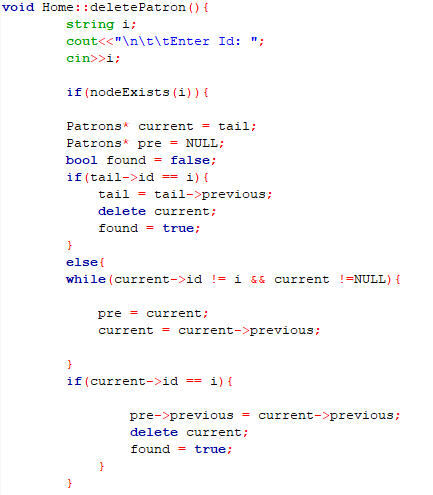


Figure 47: Delete Patron Code Snippet

# 4.0 Snippet of the working program

## 4.1 Main Manu Console

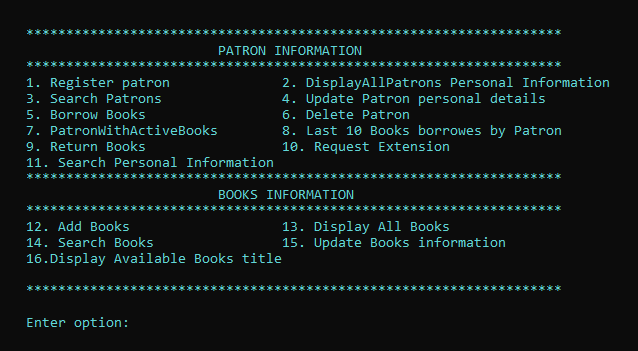


Figure 48: Main Menu Console

In the above snippet, it shows the main menu console where there are 16 options in both patron and books part. In patron there are 11 options including some extra function like return books, request extension etc.

## 4.2 Registration Patron

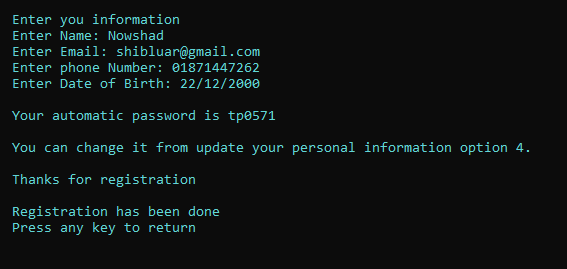


Figure 49: Registration Patron

In the figure 41 shown in the above illustrates registration of patron where during registration users need to give their name, email, phone number and data of birth. If registration become successful user will get automatic password which will be generated by program itself.

## 4.3 View Personal Details

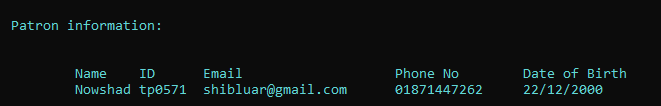


Figure 50:View Personal Details

Figure 42 in the above represent the “display personal information” function where we could see all the personal details of the patron who are in the list.

## 4.4 Add Books

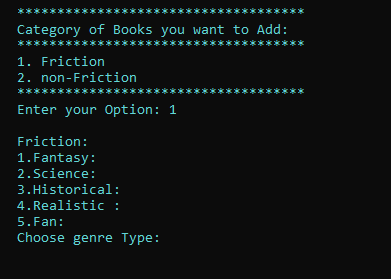


Figure 51: Option to Add Books

Add book’s function snippet is shown above where users could add any types of books depending on category and genre which is also described in the figure 44 below.

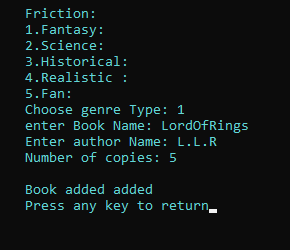


Figure 52: Adding Books

## 4.5 Borrow Books

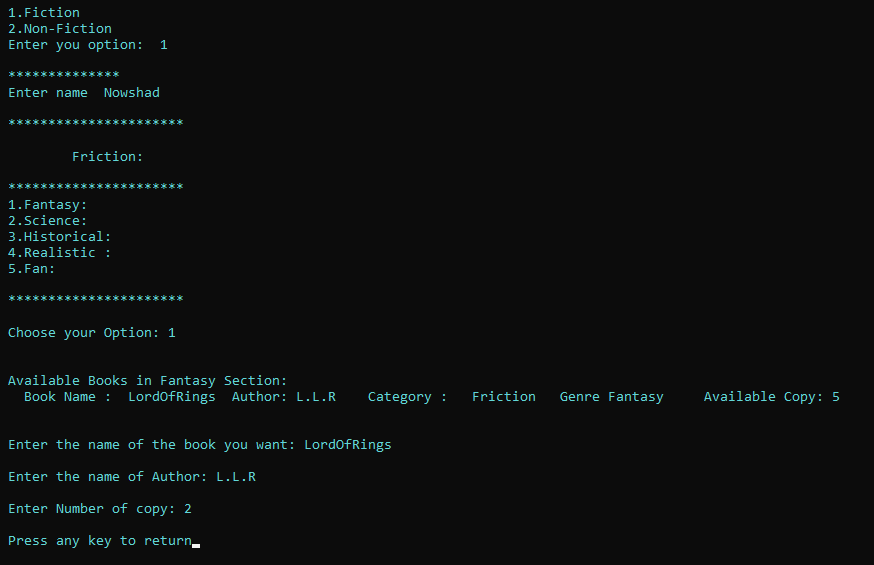


Figure 53: Borrow Books

The above snippet illustrates how a user will be taking books from the library. In order to borrow books, first user need have a confirmation with program that whether he/she is registered patron or not if he/she is in registered patron list then one can borrow books. If one borrows book they copy number will be reduced from the book list. For instance, in library there are 5 books of “LordOfRings” and if one take 2 “LordOfRings” Books than in library there will be 3 “LordOfRings” left.

## 4.6 Search patron

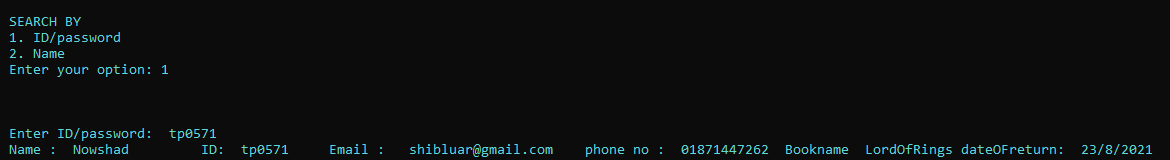


Figure 54: Search Patron

This snippet shows how one can search patron information with active book status. In this function user will be having two functions which are “search by name” and “search by ID”.

## 4.7 Updating personal Information

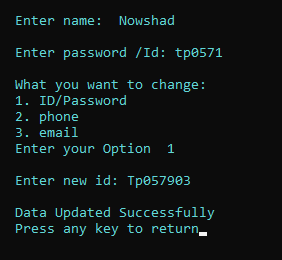


Figure 55:Update Personal Information

In figure number 47 we can see it illustrates how user can update their personal information. If anyone wants to change any information related to their ID, phone Number and Email, they can change this from the update option.

## 4.8 Patron with Active Borrowed Books

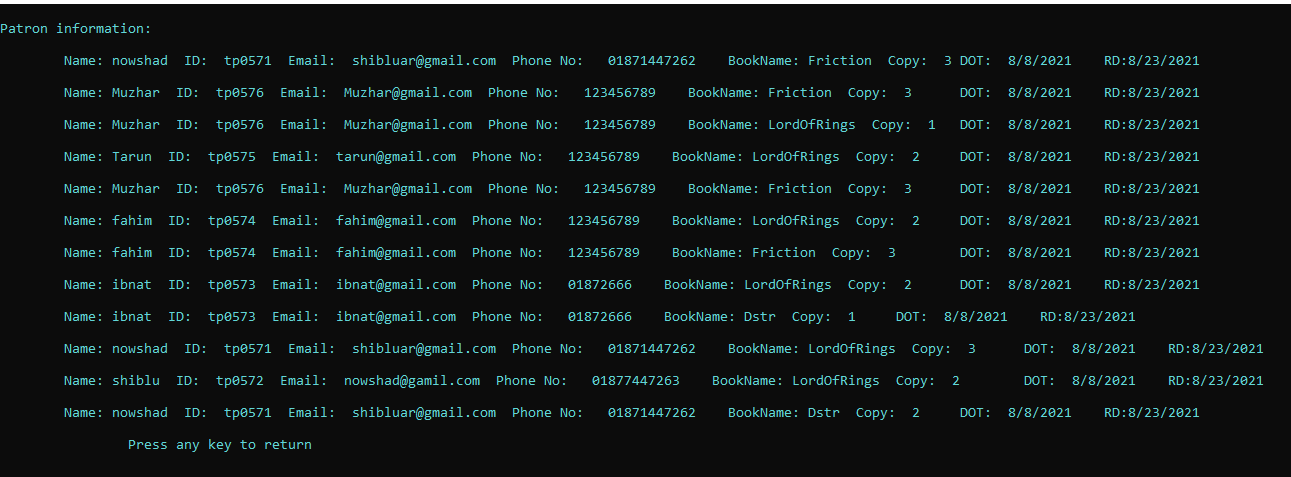


Figure 56: Patron with Active Books

The above figure where it shown all the patron information with active book status. In the function user can see patron information like name, id, email, phone number, book name, number copy, date of taken books and as well as return date. Program will automatically generate the date of taken and return which is 15 days from taken date.

## 4.9 View last 10 borrowed books

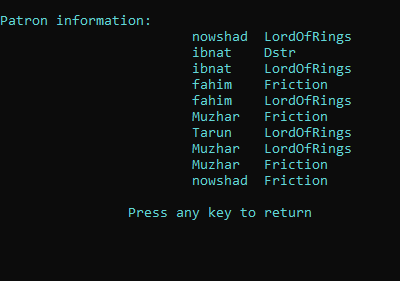


Figure 57: View Last 10 Book Borrowed by Patron

## 4.10 Return books



Figure 58: Current Books Information

In return book function user first need to enter their id and book name if it matches with the information that exist in the list then they need to enter number of copy they want to return. If their number of return book is more than the records it will not program will return to the main menu. Again, if user return any book this book will be added to the books list which is shown both in figure 50 and 52.

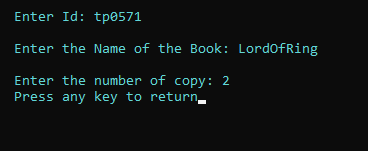


Figure 59: Returning the Books



Figure 60: Information of Books After Returning

## 4.11 Request Extension

In this option user can extend their date to return book if they want to return it after few days. To do so, users just need to enter the number of day that they want to extend program with automatically update the date which is shown in the figure 53 and 55.

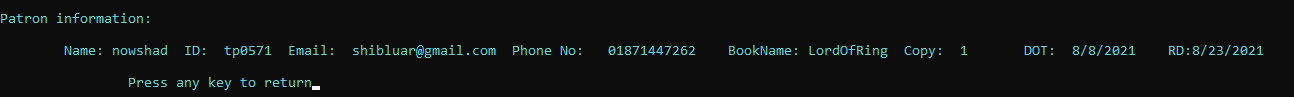


Figure 61: Current Information of Patron

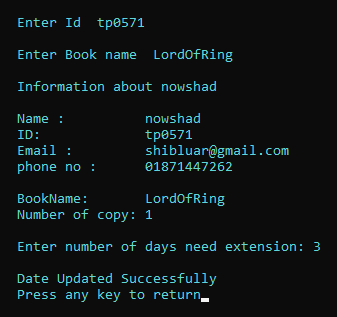


Figure 62: New Date for Extension

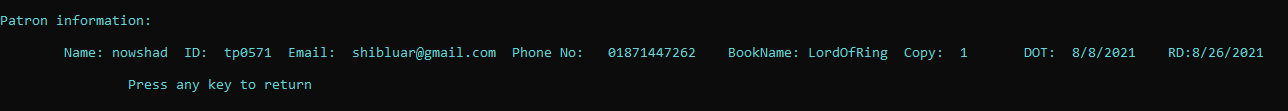


Figure 63: Information of Patron After getting Extension

## 4.12 Display all books

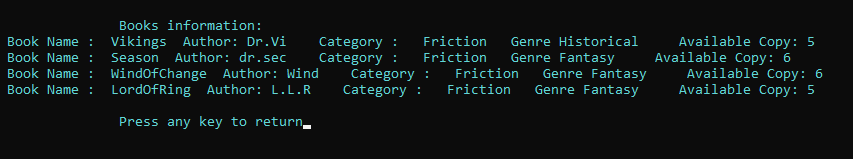


Figure 64: Display all List of the Books

## 4.13 Display available books

This is the function where users can see the available books within the library which is shown in below figure 57.

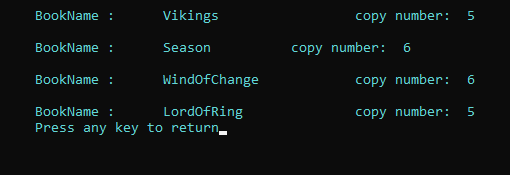


Figure 65: Display Available Title

## 4.14 Search Books

In search book function user will have the access to search book with category, genre, title and availability shown in below figure.

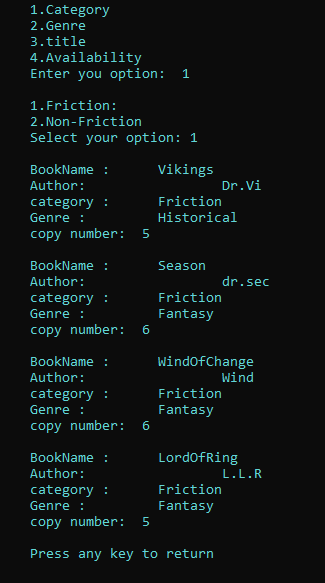


Figure 66: Search Books

# 5.0 Limitation

Though in the program that is Library management system looks quite good, yet in this console-based program there are some of the scopes where still one can improve which is search and update function. Tough trough linked list inserting and deleting can be easy, it has more time complexity when it comes to search and updating node as it takes minimum time complexity log(n). But if we use Hash table instead of linked list one can get rid from this problem as with hash table search will take time complexity of log (1). Again, as I used “cin” command for storing and scanning string it would take a single word but, in that case, if I used to get line command, we can remove this issue from this program. And last but not the least while patron borrow books using linked list every time it creates a new node for that if we could store different books and time in one node, we may have good space complexity.

# 6.0 Matrix

|  |  |  |
| --- | --- | --- |
| Task | Nowshad ul Alam | Khadija Fairus |
| Proposal | Algorithm, Visualization, description patron and Books 70% | Justification 30% |
| Implementation | 100% |  |
| Report |  |  |

# 7.0 References

DEVMOUNTAIN, 2021. *what you Should know about data structure.* [Online]   
Available at: https://blog.devmountain.com/what-you-should-know-about-data-structures/  
[Accessed 2 July 2021].

GeeksforGeeks, 2021. *Data Structures.* [Online]   
Available at: https://www.geeksforgeeks.org/data-structures/  
[Accessed 2 july 2021].

GeeksforGeeks, 2021. *Doubly Linked List | Set 1 (Introduction and Insertion).* [Online]   
Available at: https://www.geeksforgeeks.org/doubly-linked-list/  
[Accessed 10 july 2021].

Point, J. T., 2021. *Data Structure.* [Online]   
Available at: https://www.geeksforgeeks.org/doubly-linked-list/  
[Accessed 2 July 2021].

TANNER, L., 2021. *Advantages Of Data Structure.* [Online]   
Available at: https://www.enotes.com/homework-help/what-advantages-disadvantages-data-stucture-479073  
[Accessed 2 jULY 2021].

TUTORIALSPOINT, 2021. *Data Structure - Doubly Linked List.* [Online]   
Available at: https://www.tutorialspoint.com/data\_structures\_algorithms/doubly\_linked\_list\_algorithm.htm  
[Accessed 2 july 2021].