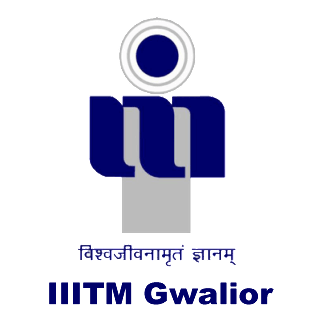
ABV - Indian Institute Of Information Technology And Management, Gwalior



*Library Management System*

Subject: Object Oriented Programming Language

|  |  |
| --- | --- |
| ***Submitted to:*** | ***Submitted by:*** |
| ***Dr. Santosh Singh Rathore*** | ***Lakshay Parashar (2020IMG-033)***  ***Mrunal Badole (2020IMG-039)***  ***Sohan Bandary (2020IMG-016)***  ***Pragya Rathore (2020IMG-049)*** |

*Contents*

[Introduction 3](#_Toc75715790)

[Working 4](#_Toc75715791)

[Books Functions 4](#_Toc75715792)

[***Add a New Book*** 4](#_Toc75715793)

[***Remove a Book*** 5](#_Toc75715794)

[***Display Book List*** 7](#_Toc75715795)

[Student Functions 8](#_Toc75715796)

[***Add new Account*** 9](#_Toc75715797)

[***Removing an Account*** 11](#_Toc75715798)

[***Display Function*** 12](#_Toc75715799)

[***Student Information*** 13](#_Toc75715800)

[***Issuing A Book*** 15](#_Toc75715801)

[***Prime Member Example*** 18](#_Toc75715802)

[***Returning A Book*** 19](#_Toc75715803)

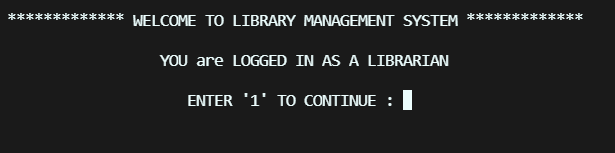
[***Final Notes Regarding Code*** 22](#_Toc75715804)

[Project Work Distribution 22](#_Toc75715805)

# *Introduction*

Our Library Management System's major goal is to manage all of the main functions of any books library, whether it's a private, public, university, or national library. This system will keep track of the books in the library, as well as the details of the books, the books that have been issued out, and the student’s records. The program will also keep track of how many books each student has and impose a restriction on how many books can be taken. The system will also place a restriction on how long a student may keep the book. Each day when the time limit is exceeded, a fine will be imposed. We also have the notion of providing pupils with unique ID numbers that correspond to their ID cards. A separate portion of our system allows you to add new books from various publishers. It also has a section for recording the records of the library's premium members, who will receive additional perks.

Currently Our program works as a librarian only system due to lack of knowledge about file handling and way to store member , so it starts off as below



Rest of the working is explained below

# *Working*

When from the login screen you enter out System , we provide you with 2 major options to operate on , namely

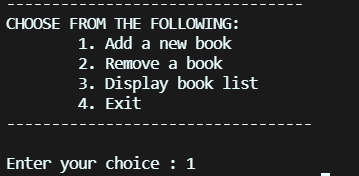
1. Students
2. Books

So lets start by adding some books because of course it’s a library after all

## *Books Functions*

So on the first option after entering into the book section we provide you with various option regarding books weather to

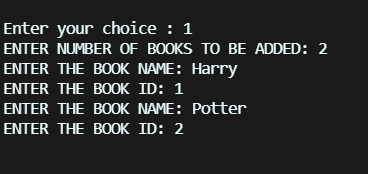
1. Add a New book
2. Remove a Book
3. Display whole Book list



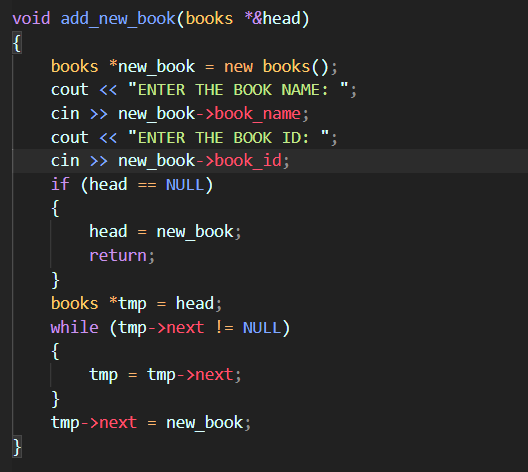
Lets go through them one by one.

### ***Add a New Book***

For this particular option we first ask how many books are to be added at a single time by the user and thus applied a For loop and implemented the Following Function named **Add\_New\_Book** Which can be referred below to add books. We can store books with Book name and Book Id so that it is easier for us to navigate through many books .

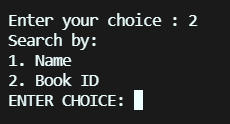


Function



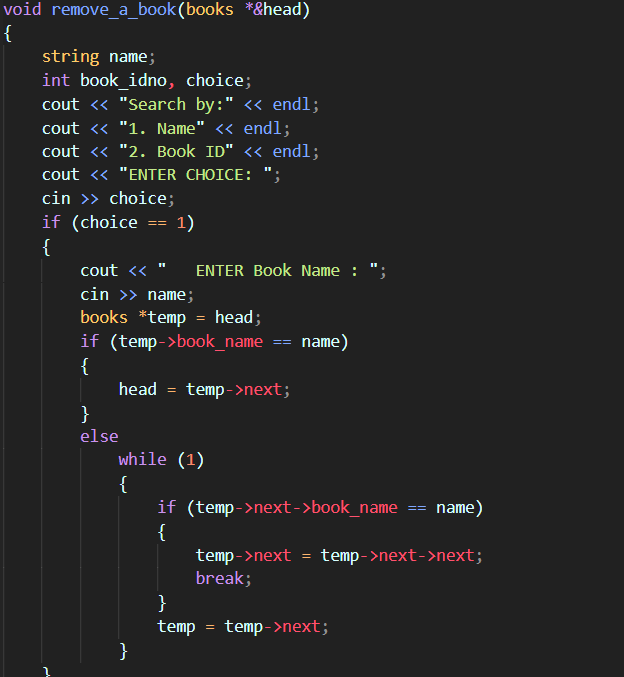
### ***Remove a Book***

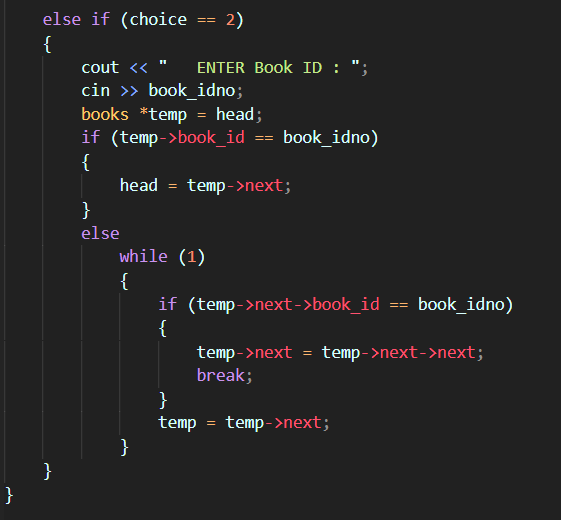
As Seen Below , we can browse through books by their name or ID , which was mentioned above



What it does is it will remove a book from the book list we have , it is helpful for a old book or a book that has been lost in the library

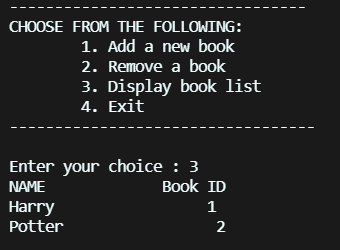
You can understand this procedure by seeing the function code Below





### ***Display Book List***

Now as the name suggests , it will display all the books we have in out list in a Tabular format, you can refer the book\_display function below the output for understanding

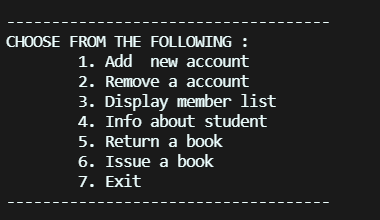




Now as all the part is book functions are discussed lets move on to Function for students

## *Student Functions*

As soon as we enter Student section we are again as seen in Books, are promoted with Various list of functions as shown below



So lets Move on from here one function by one.

### ***Add new Account***

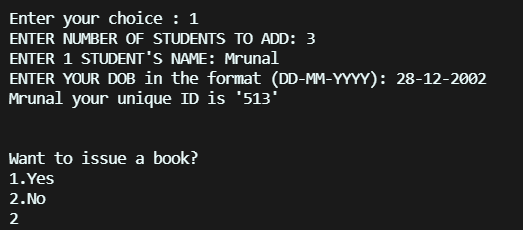
As the name suggests it allows the librarian to add a student account so they student can have their own id and issue / return the book

Here also we have Implemented a for loop to add multiple account at the same time , it can be referred as the code and output below

We will add a few demo Accounts

On this step , we will ask for various details about the students like their name and Date of Birth , using this and combining this we will make a Unique Id of Students That will be useful While Issuing a book or retuning a book

NOTE: we also added a functionality of issuing a book At the time of making a account but we will see that later in the report



Code for the Add Student part is as follows

void add\_new\_student(students *\*&*head, int i)

{

    int choice;

    students \*new\_student = new students();

    cout << "ENTER"

         << " " << i << " STUDENT'S NAME: ";

    cin >> new\_student->student\_name;

    cout << "ENTER YOUR DOB in the format (DD-MM-YYYY): ";

    cin >> new\_student->dob;

    new\_student->unique\_id = new\_student->student\_name[0] + new\_student->student\_name[1] + new\_student->student\_name[2] + new\_student->dob[0] + new\_student->dob[1] + new\_student->dob[3] + new\_student->dob[4];

    cout << new\_student->student\_name << " your unique ID is "

         << "'" << new\_student->unique\_id << "'" << endl;

    cout << endl;

    new\_student->prime\_id = new\_student->unique\_id;

    cout << endl;

    cout << "Want to issue a book?" << endl

         << "1.Yes" << endl

         << "2.No" << endl;

    cin >> choice;

    if (choice == 1)

    {

        issue\_book(new\_student, choice);

    }

    else if (choice == 2)

    {

        if (head == NULL)

        {

            head = new\_student;

            return;

        }

        students \*tmp = head;

        while (tmp->next != NULL)

        {

            tmp = tmp->next;

        }

        tmp->next = new\_student;

        return;

    }

    else

    {

        cout << "INPUT IS INVALID";

        exit(0);

    }

    if (head == NULL)

    {

        head = new\_student;

        return;

    }

    students \*tmp = head;

    while (tmp->next != NULL)

    {

        tmp = tmp->next;

    }

    tmp->next = new\_student;

}

### ***Removing an Account***

This Function is also very useful weather if the librarian want to remove a particular students account as all its work is done.

Here Also we give option for the librarian to browse through name or Student’s Unique Id.

Can be explained much better with the code below

void remove\_a\_student(students *\**head)

{

    string name;

    int unique\_idno, choice;

    cout << "Search a member by:" << endl;

    cout << "1. Name" << endl;

    cout << "2. Unique ID" << endl;

    cout << "ENTER CHOICE: ";

    cin >> choice;

    if (choice == 1)

    {

        cout << "   ENTER YOUR NAME: ";

        cin >> name;

        students \*temp = head;

        if (temp->student\_name == name)

        {

            head = temp->next;

        }

        else

            while (1)

            {

                if (temp->next->student\_name == name)

                {

                    temp->next = temp->next->next;

                    break;

                }

                temp = temp->next;

            }

    }

    else if (choice == 2)

    {

        cout << "   ENTER THE UNIQUE ID: ";

        cin >> unique\_idno;

        students \*temp = head;

        if (temp->unique\_id == unique\_idno)

        {

            head = temp->next;

        }

        else

            while (1)

            {

                if (temp->next->unique\_id == unique\_idno)

                {

                    temp->next = temp->next->next;

                    break;

                }

                temp = temp->next;

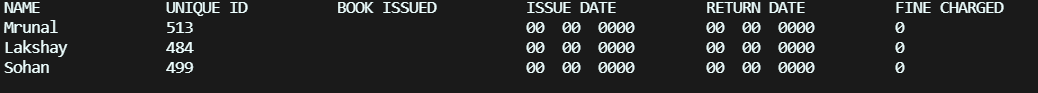
            }

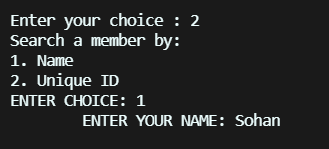
    }

}

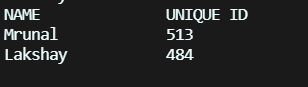
Suppose we Remove one student by name

Before:





After:



### ***Display Function***

You may wonder , the above Example, yes they are the format in which we display the information about the Students whose account are made in the system ,

void display(students *\**head)

{

    int i;

    if (head == NULL)

    {

        cout << "NO LIST AVAILABLE";

        return;

    }

    students \*tmp = head;

    cout << "NAME"

         << "              "

         << "UNIQUE ID"

         << "          "

         << "BOOK ISSUED"

         << "          "

         << "ISSUE DATE"

         << "          "

         << "RETURN DATE"

         << "          "

         << "FINE CHARGED" << endl;

    while (1)

    {

        cout << tmp->student\_name;

        for (i = 0; i < 18 - (tmp->student\_name.length()); i++)

            cout << " ";

        cout << tmp->unique\_id;

        cout << "                ";

        cout << tmp->name\_issued\_book;

        for (i = 0; i < 21 - (tmp->name\_issued\_book.length()); i++)

            cout << " ";

        printf("*%.2d*  *%.2d*  *%.4d*", tmp->day\_issued, tmp->month\_issued, tmp->month\_issued);

        cout << "        ";

        printf("*%.2d*  *%.2d*  *%.4d*", tmp->day\_returned, tmp->month\_returned, tmp->month\_returned);

        cout << "         ";

        fine\_charging(tmp);

        cout << tmp->fine\_charged << endl;

        if (tmp->next == NULL)

            return;

        else

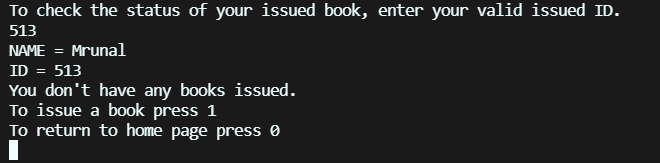
            tmp = tmp->next;

    }

}

### ***Student Information***

But let suppose we want all information about a particular student, how will that be shown , Yes this particular function is there for it , here is the one more use of Unique Id We made Before while making student accounts.We need to add the Unique ID in order to get Information about the Students.



As Seen above We haven’t already issued a book , but here again it ask us to issue to book if we haven’t already , just like when we were making a Account.

void check\_info(students *\**tmp)

{

    int ID, choice;

    cout << "To check the status of your issued book, enter your valid issued ID." << endl;

    cin >> ID;

    while (1)

    {

        if (tmp->prime\_id == ID)

        {

            cout << "NAME = " << tmp->student\_name << endl;

            cout << "ID = " << tmp->prime\_id;

            cout << endl;

            if (tmp->book\_ID == 0)

            {

                cout << "You don't have any books issued." << endl;

                cout << "To issue a book press 1" << endl

                     << "To return to home page press 0" << endl;

                cin >> choice;

                if (choice == 1)

                {

                    issue\_book(tmp, choice);

                }

                else if (choice == 0)

                {

                    return;

                }

                else

                {

                    cout << "INPUT IS INVALID";

                    exit(0);

                }

            }

            else

            {

                cout << "You already have a book issued" << endl;

                cout << "BOOK NAME = " << tmp->name\_issued\_book << endl;

                cout << "BOOK ID = " << tmp->book\_ID << endl;

                cout << "Issued date = " << tmp->day\_issued << " " << tmp->month\_issued << " " << tmp->year\_issued << endl;

                cout << "The number of days for which the user has issued the book = " << tmp->number\_days << endl;

            }

        }

        else

        {

            tmp = tmp->next;

        }

    }

}

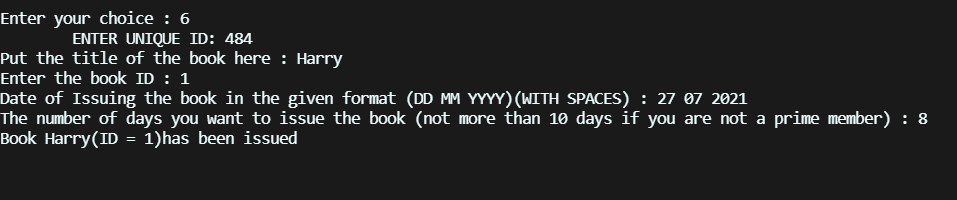
As seen in the code , If we had a book , it will show us information about issue transaction. Our next step is that itself , Finally We will Issue a book

### ***Issuing A Book***

As We enter the Issue book function in any Condition it will ask us the Unique ID of the student , so we can update the information about that student.

When issuing a book it will ask us information as Book name , Book Id , The Date of Issue , and for how many Days we want to issue a book

It will Output the Following Information



void issue\_book(students *\*&*new\_student, int choice)

{

    int choice1;

    switch (choice)

    {

    case 1:

        cout << "Put the title of the book here : ";

        cin >> new\_student->name\_issued\_book;

        cout << "Enter the book ID : ";

        cin >> new\_student->book\_ID;

        {

            int date\_check = 1;

            while (date\_check)

            {

                cout << "Date of Issuing the book in the given format (DD MM YYYY)(WITH SPACES) : ";

                cin >> new\_student->day\_issued >> new\_student->month\_issued >> new\_student->year\_issued;

                if (new\_student->month\_issued == 1 || new\_student->month\_issued == 3 || new\_student->month\_issued == 5 || new\_student->month\_issued == 7 || new\_student->month\_issued == 8 || new\_student->month\_issued == 10 || new\_student->month\_issued == 12)

                {

                    if (new\_student->day\_issued > 31)

                        cout << "INVALID DATE" << endl;

                    else

                        date\_check = 0;

                }

                else if (new\_student->month\_issued == 11 || new\_student->month\_issued == 6 || new\_student->month\_issued == 9 || new\_student->month\_issued == 11)

                {

                    if (new\_student->day\_issued > 30)

                        cout << "INVALID DATE" << endl;

                    else

                        date\_check = 0;

                }

                else if (new\_student->month\_issued == 2)

                {

                    if (((new\_student->year\_issued % 4 == 0) && (new\_student->year\_issued % 100 != 0)) || (new\_student->year\_issued % 400 == 0))

                        date\_check = 0;

                    else

                        cout << "INVALID DATE";

                }

            }

        }

        cout << "The number of days you want to issue the book (not more than 10 days if you are not a prime member) : ";

        cin >> new\_student->number\_days;

        if (new\_student->number\_days <= 10 && new\_student->number\_days > 0)

        {

            cout << "Book " << new\_student->name\_issued\_book << "(ID = " << new\_student->book\_ID << ")"

                 << "has been issued" << endl;

        }

        else if (new\_student->number\_days > 10)

        {

            if (new\_student->prime\_id == new\_student->unique\_id + 80)

            {

                return;

            }

            else

            {

                cout << "You are not a member of our library's prime group." << endl

                     << "If you want to be a part of it press 1" << endl

                     << "1.Yes" << endl

                     << "2.No" << endl;

                cin >> choice1;

                if (choice1 == 1)

                {

                    add\_prime\_member(new\_student);

                }

                else if (choice1 == 2)

                {

                    return;

                }

                else

                {

                    cout << "INVALID INPUT";

                    exit(0);

                }

            }

        }

        {

        }

        break;

    case 2:

        break;

    default:

        cout << "INPUT IS INVALID";

        exit(0);

        break;

    }

}

As Seen in the Code , if we try to issue a book for more than 10 days , it will Prompt us with a error saying ‘ You are not a Prime member of our library’

We have Implemented this Prime member check For almost all functions in our system.

Here comes the Most interesting feature of our system , that is a PRIME MEMBER,

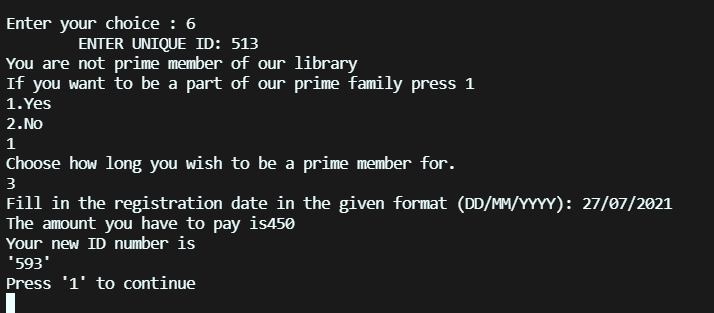
The Benefits of Being Prime is that

1. we can issue a book for more that 10 days
2. No Fine what-so-ever
3. Can issue Multiple Books

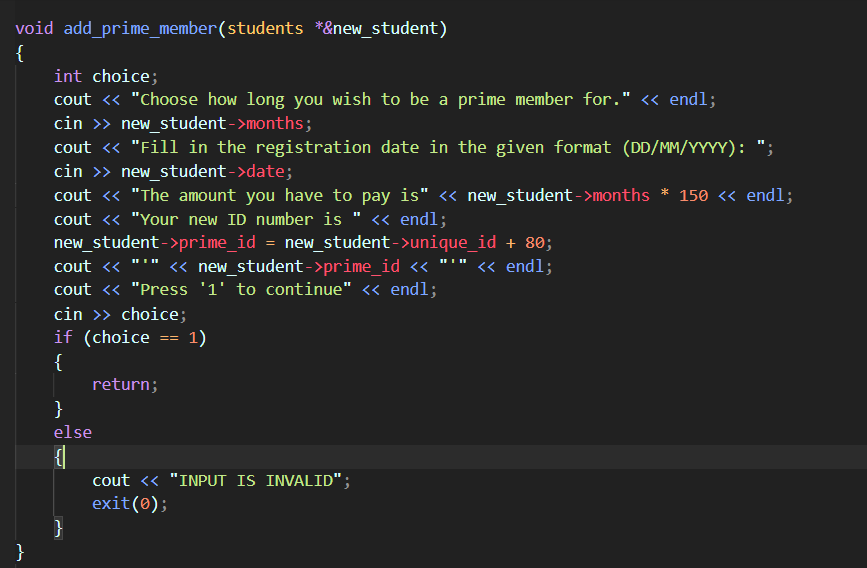
NOW Suppose we try to be a Prime member , it will Ask us Certain Important information and Payment receipt of being a prime member ,

It can seen as below.

### ***Prime Member Example***



And this is How we add a prime Member

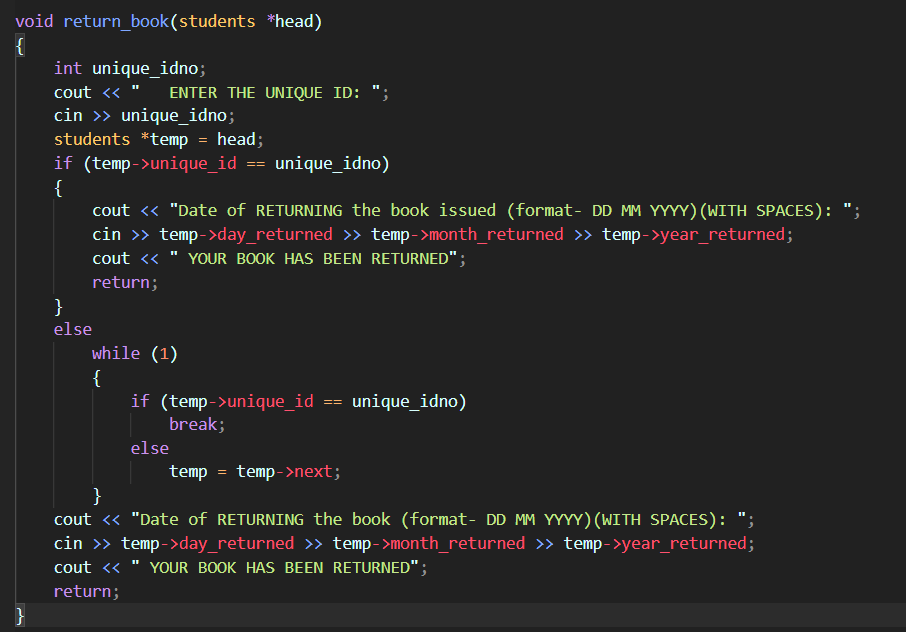


After this , we can Issue a book for more than 10 days , return it anytime and bypass the fees Payment of Late Return

Now Lets get to the Final Part of the Syste, that is Returning a Book.

### ***Returning A Book***

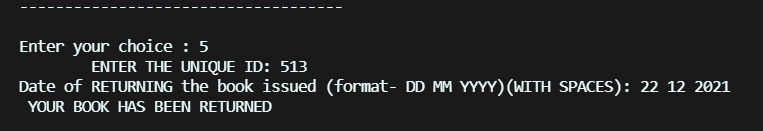
As other Functions , this will also ask the librarian a few Things bout the student is , returning date , and other necessary information



First We will, return a book of a prime member , with more than 10 days later than that of the issuing date to give one more example of Prime Members

##### Prime Member Returning A Book

Easy and simple with no fees

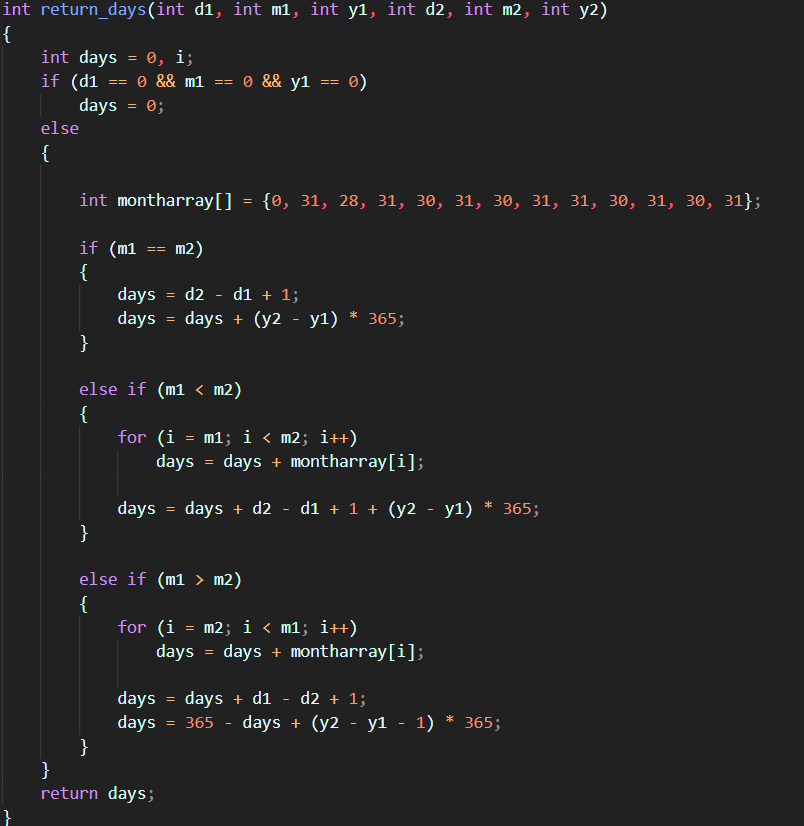


#### Non-Prime Member Returning A Book

Now Here it gets a little bit complicated , we will return after 10 days to see few left out function , which are day count function and Fine Function,

#### Date Count Function

As the name Suggest it will return the number of days the book was with the student , which will help us to determine how much fees to take from the student for late Return



#### Fine Function

We implemented a if Else to take 0 fees from Prime Members , and for regular members, we charge Rupees 2 per each delayed day , can be changed in code

void fine\_charging(students *\*&*stud)

{

    if (stud->unique\_id == stud->prime\_id)

    {

        int number\_of\_days = return\_days(stud->day\_issued, stud->month\_issued, stud->year\_issued, stud->day\_returned, stud->month\_returned, stud->year\_returned);

        if (number\_of\_days != 0)

        {

            stud->fine\_charged = (number\_of\_days - stud->number\_days) \* 2;

            if (stud->fine\_charged < 0 || number\_of\_days == 0)

                stud->fine\_charged = 0;

            return;

        }

        else

            return;

    }

}

### ***Final Notes Regarding Code***

So that was it for all for the Functionality of the Library management system.

There Still can be come Bugs found which we might’ve missed , and also some pre-planned functionalities are missing due to lack of ideas to implement and knowledge , for example having a data base to store all the Data even when a session ends.

The System is Completely based on dynamic memory so if it gets Bugged out in corner cases all the stored data is lost. This Could be Solved with a proper database Management.

# *Project Work Distribution*

As for the distribution of the project , each and every member did a equally good amount of work

As for the idea , Lakshay came up with this management , which was later modified by Sohan With added Functionalities.

Book part was Coded by Pragya and Me(Mrunal) along with some Interrelated code about student and Books

Rest of the Student part was Handled by Sohan and Lakshay , and Finally Some final Touches were added by me and Lakshay. So As Seen Every Member was equally important in this project.

We Hope you could understand Working and other processes of the project , or Even if you didn’t understand we’ve added the Project Code File along with it to Make You fully Satisfied with the project.

We hope you liked it

Thank You