**LIBRARY MANAGEMENT SYSTEM**

**PROJECT REPORT**

GROUP NO:7

*Submitted by,*

19Z305 - Akshara. P

19Z311 - Dhanavandhana.K

19Z345 - Saranya.K

19Z343 - Samyuktha.A.S.K

19Z352 - Sushmitha. S

**ABSTRACT**

Libraries today are fast growing and management today key has a problem. This project aims in developing a computerized system to solve this problem and maintain all daily works in a library. This project developed using the C programming language has many features which are not available in a normal library management system like facility of librarian login and students’ login. It allows the users to maintain a record of books and retrieve them on command. It mainly focuses on the basic operations in a library like adding new books, updating student information, searching books and the facility to borrow and return books. Overall, this project is developed to help students and the librarian to maintain the library in the best way possible with reduced human efforts.

**PROBLEM STATEMENT:**

To develop and implement a system that can handle and manage the activities involved in a library in an efficient and reliable way.

The objectives of the project are as follows:

* To build a system that can perform library management operations efficiently.
* To store and provide easy access to accurate data.
* To increase productivity and reduce costs in library management.
* To reduce errors and to eliminate repetitive manual processing in the library.

**EXISTING SYSTEM:**

Presently, transaction of books in libraries are carried out manually. A large number of registers are maintained containing the list of students and their details, list of books and their details. Students wait in lines to sign out books manually with cue cards indicating their names. Other transactions like searching of books, borrowing books, returning them are also carried out manually resulting in time consuming inefficient library management.

**DRAWBACKS OF EXISTING SYSTEM:**

The following are the challenges faced by the users while dealing with manual library management system.

* Poor data storage leading to loss of data.
* Slow retrieval of data thus wasting time.
* Lack of data storage.
* Lack of security.
* Misplaced memory cards.
* Errors in maintenance.
* Decreased efficiency and accuracy.
* Expensive.

**PROPOSED SYSTEM:**

Based on the difficulties sited above, it is required that the new system targets the weakness of the old system in place. This requirement resulted in the development of the proposed system with the following solutions:

* Provide better and efficient service to users.
* Reduce the workload of librarian.
* Faster retrieval of information.
* Reduce the possibility of error.
* Improve productivity and reduce costs.
* Reduce the duplication of data.

The proposed system is an automated library management system. Through the proposed system one can add books, search books, update information, delete information and many such operations in quick time. All the manual difficulties encountered managing the library have been rectified by computerization.

**SYSTEM DESIGN:**

The programming language used to develop the system is the C programming language. The system can be used by both students and the librarian. The users are directed to login or signup in order to access the menu page. The menu page provides access to features depending on whether the user is the librarian or a student.

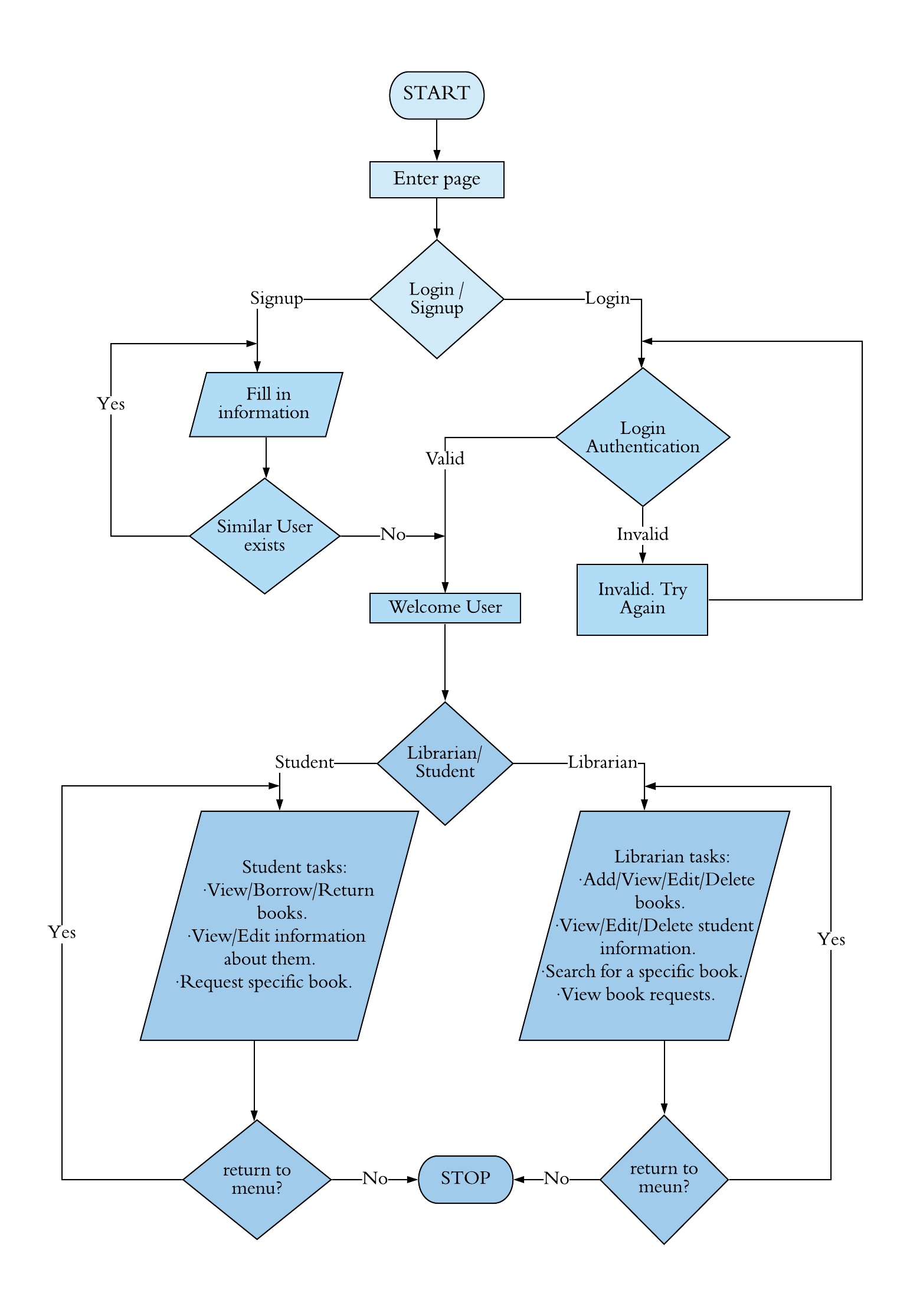
Librarian features:

* Add/View/Edit/Delete books.
* View/Edit/Delete student information.
* Search for a specific book.
* View book requests.

Student features:

* View/Borrow/Return books.
* View/Edit information about them.
* Request specific book.

Once a particular task is complete the users can either return to the menu page or exit from the software.

Pictorial representation of the system is given in the below flowchart: 

**IMPLEMENTATION:**

**Requirements:**

1. 800 MHz processor or above
2. RAM- 512 MB or larger
3. Minimum 20 MB of hard disk space
4. Operating System – Windows, Linux, mac
5. Additional software required – Codeblocks

**Assumptions and dependencies:**

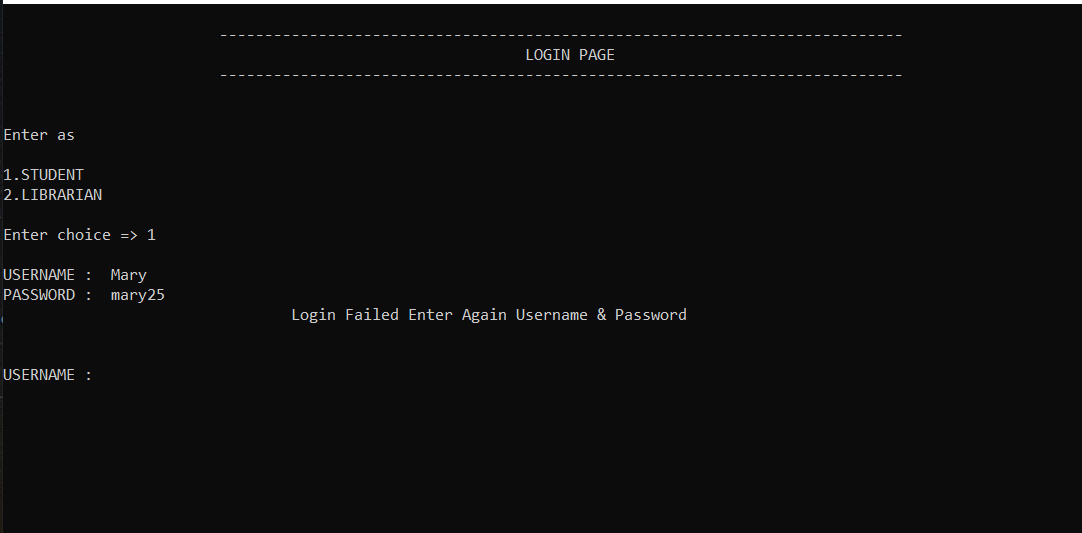
It is assumed that all data given as input are of the corresponding data type used and without spaces.

**IMPLEMENTATION DETAILS OF EACH MODULE:**

**1. login( ):**

The function allows the user to identify himself before entering the computerized system. The user can either login as a librarian or a student. It requests the user for his username and also his password in order to check whether the person has an existing account and is authorized to use the system.

**OUTPUT:** On success: goes to menu page.On failure:



**2. sign\_up( ):**

It allows students to create a new account for themselves in the system. It makes sure every student has a unique id and username. The function asks additional information from the user and stores them in the file containing student data.

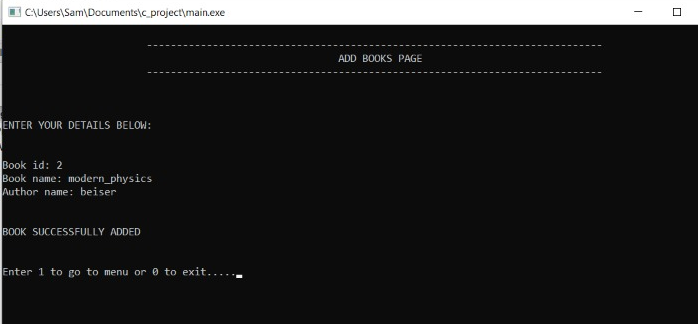
**OUTPUT:**



**3. add\_book( ):**

It allows user (Librarian) to add new books to the library. This function add books and its details (like book id, book name, author name) in a file and it displays ‘Book successfully added’. The record is saved in a file.

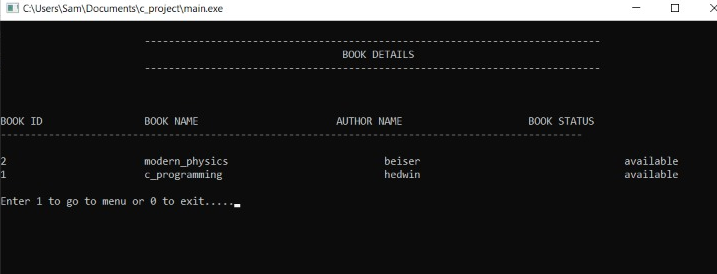
**OUTPUT:**



**4. view\_books( ):**

This function displays the details of books added (stored), including the status of the stored books. If the book is not available in the record, then it displays “No record”.

**OUTPUT:**

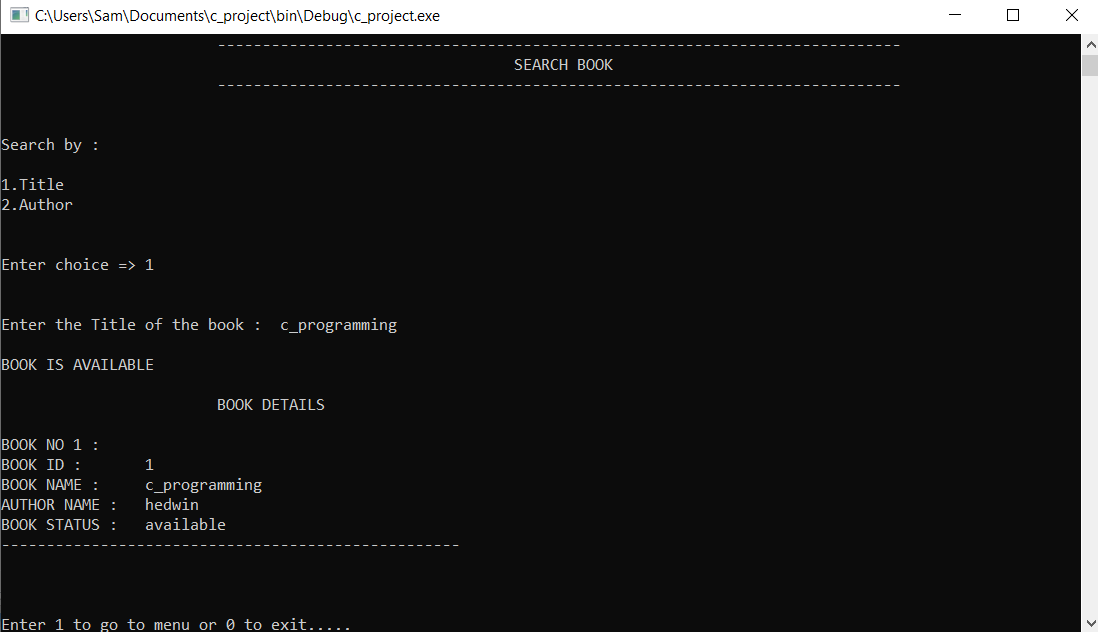


**5. search\_book( ):**

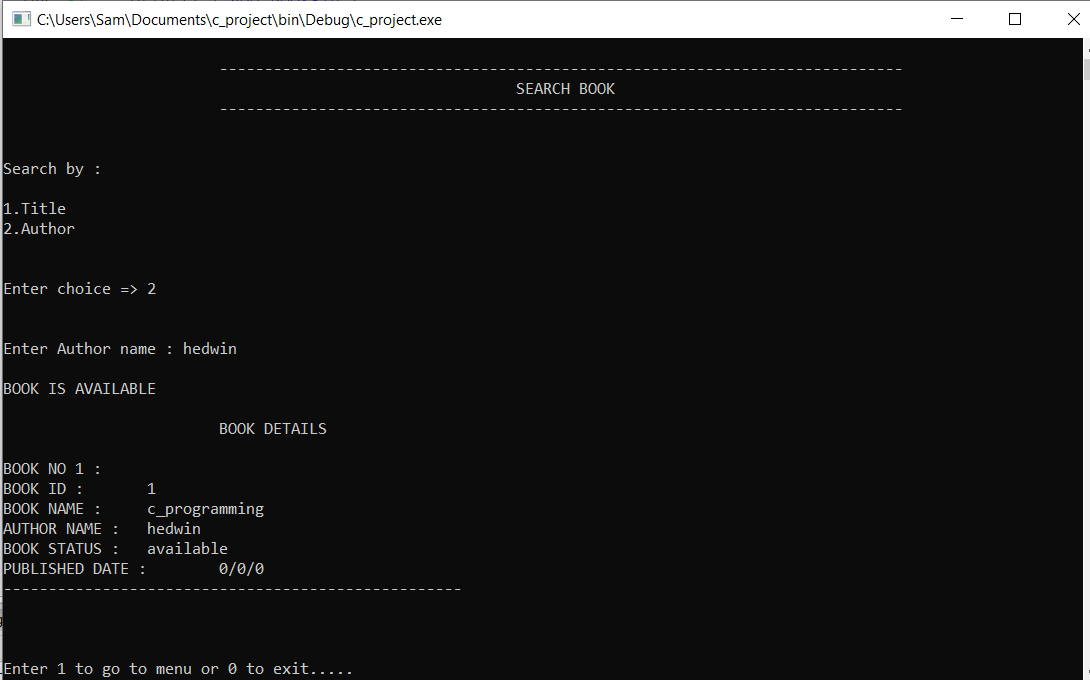
The user can search for a specific book in the library using this search book function. It searches books in two means, either using the book’s title or the author’s name. This function is of void type and has no arguments**.**

**OUTPUT:**

If choice ==1



If choice==2

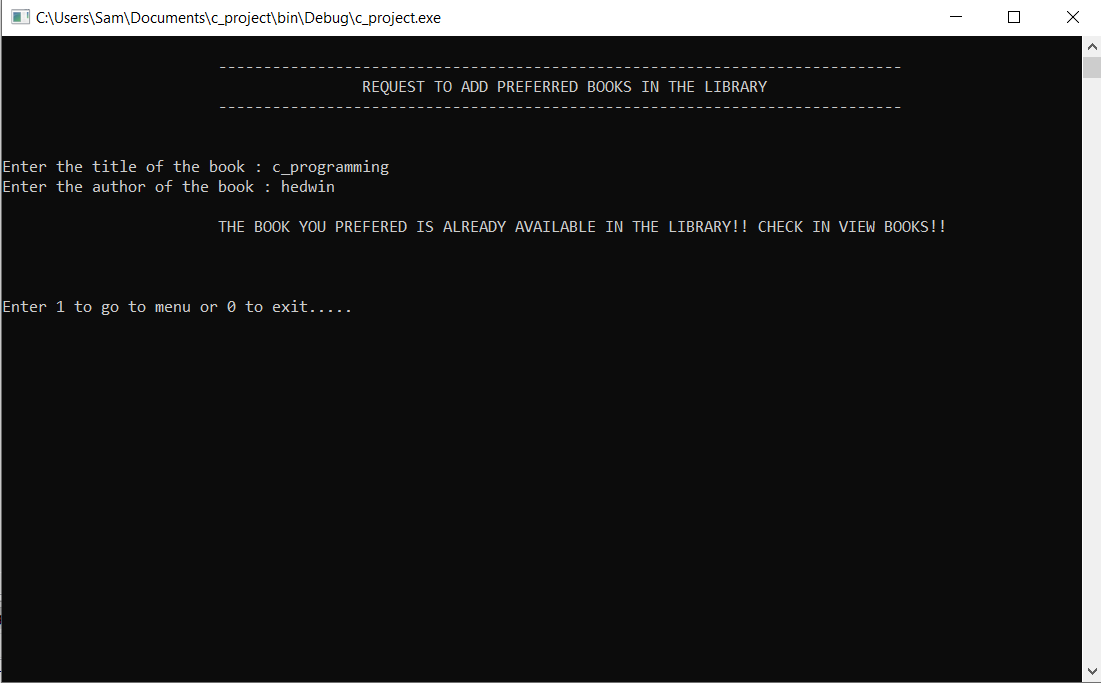


**6.** **add\_preference ():**

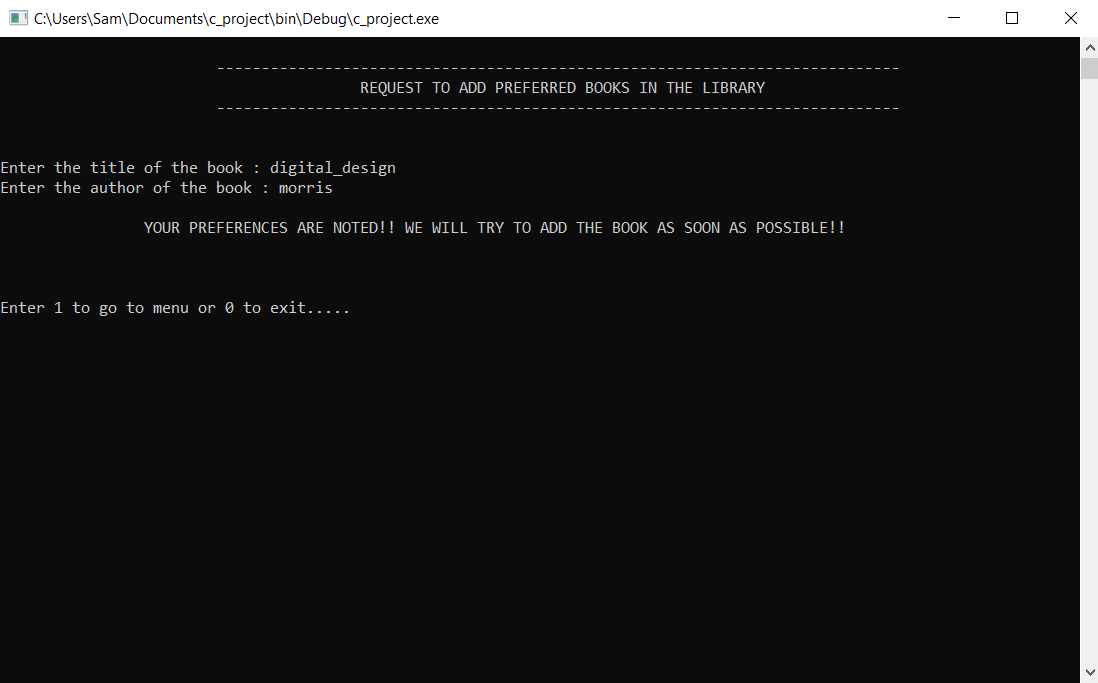
The user can prefer books to the library using this add\_preference function. Here, the preferred book’s details (name of the title and author’s name) are entered and are added in a file which can be later viewed by the librarian. This function is of void data type and has no arguments.

**OUTPUT:**

If the book is already available:



if the book is not available in the library:

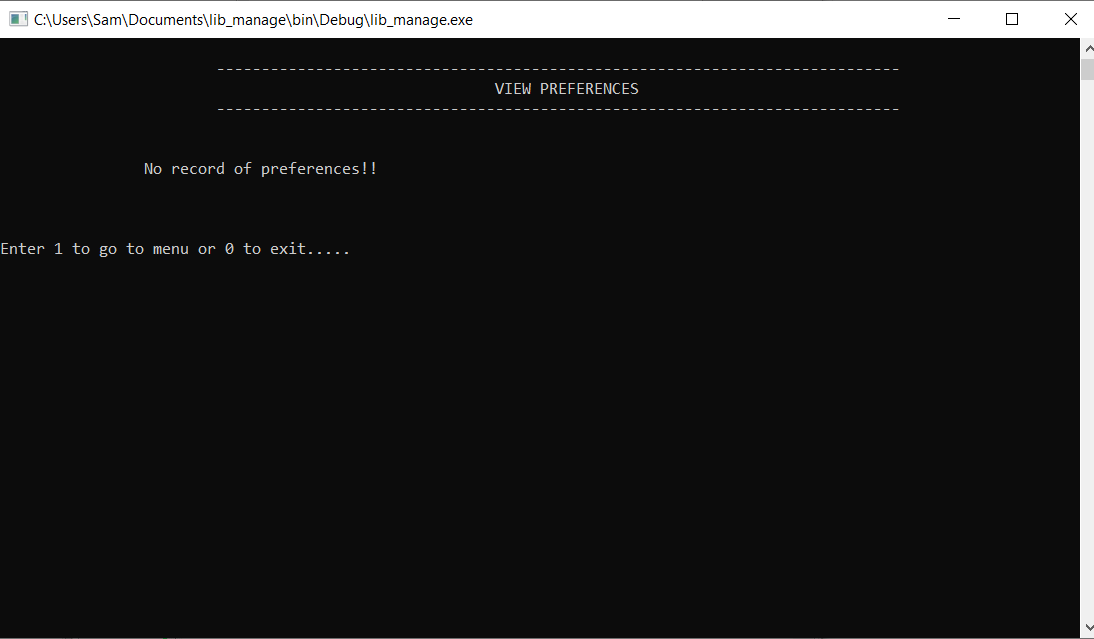


**7.** **view\_preference ():**

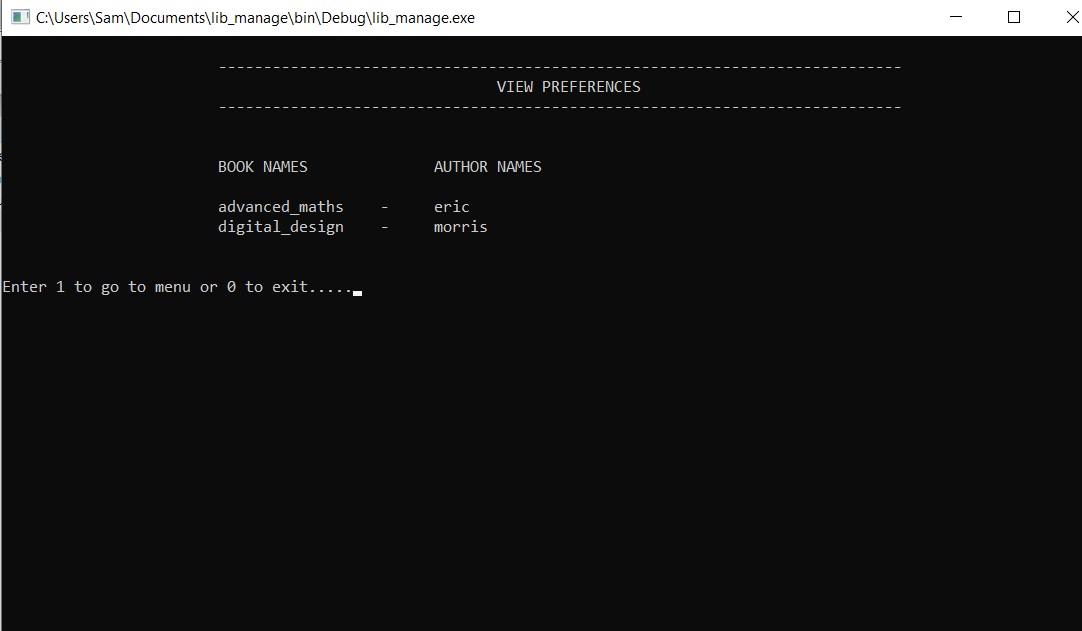
This function allows the librarian to view the details of the books preferred by the students. It prints preferred books details like book’s title and author’s name on the console. This function is of void type and has no arguments.

**OUTPUT:**

If the file pointer is NULL:



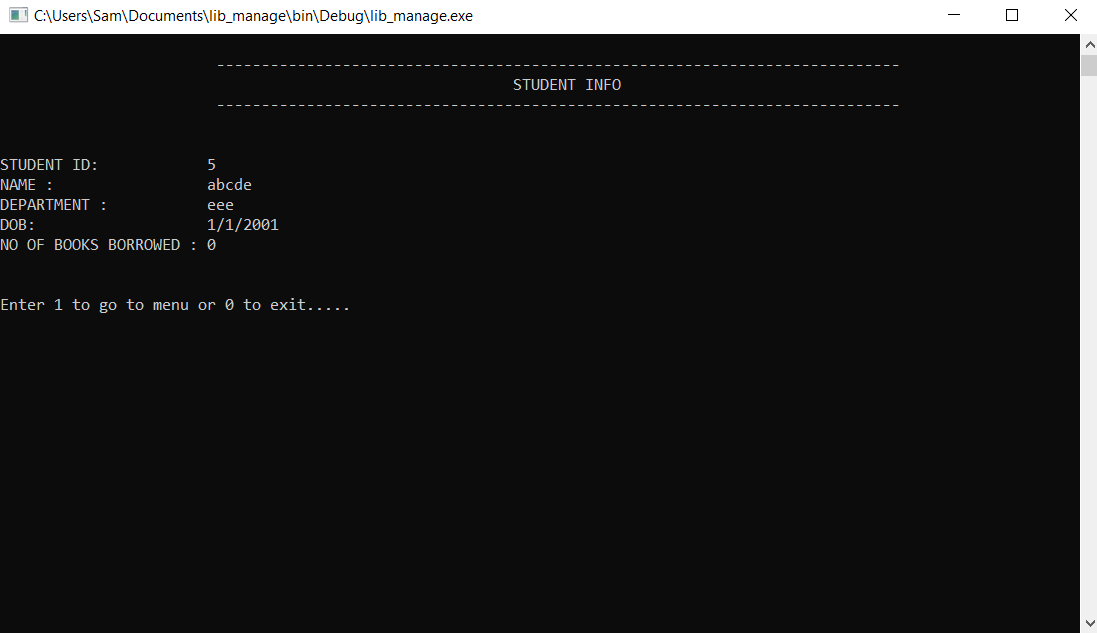
If the file pointer is void:



**8.** **display\_info ():**

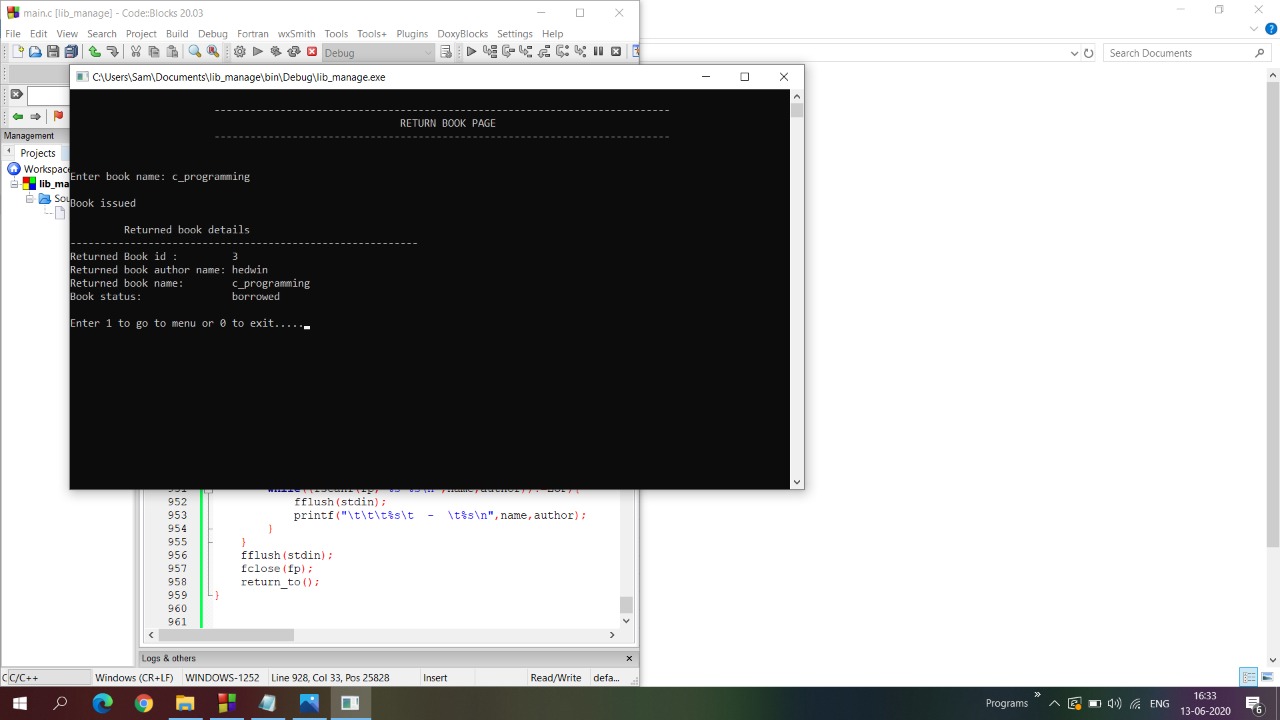
This function shows the current details of the user. Student details like id\_no, name, department, DOB, etc are printed on the terminal. This function is of void data type and has no arguments**.**

**OUTPUT:**



**9. return\_book ():**

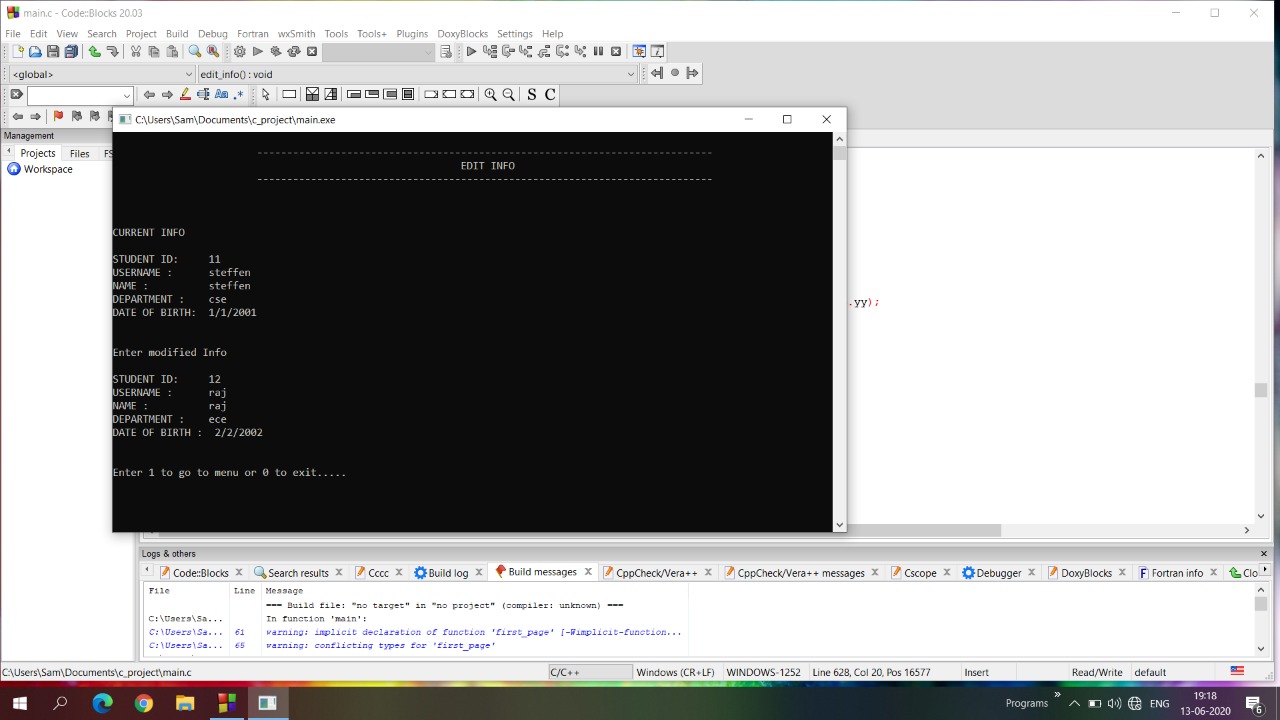
This function allows students to return books that they borrowed earlier and then returned book is removed from student’s borrowed books record.

**OUTPUT:** 

**10. edit\_info ():**

This function allows the librarian and student to modify the current information of student like student Id, username, name, department, date of birtand these modifications are updated.

**OUTPUT:**

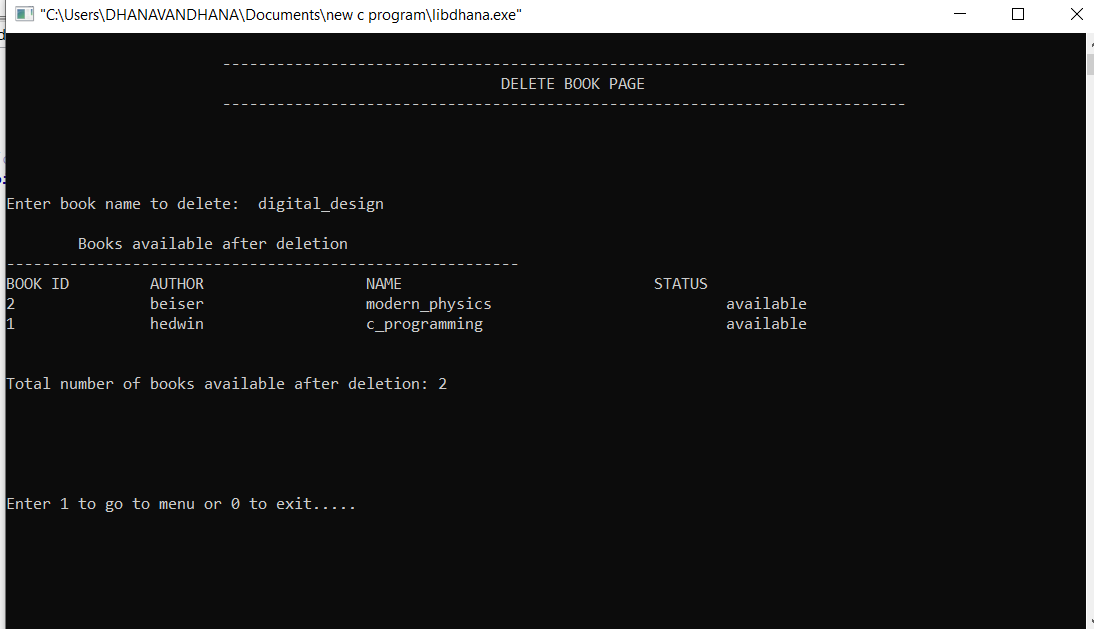


**11. delete\_book ():**

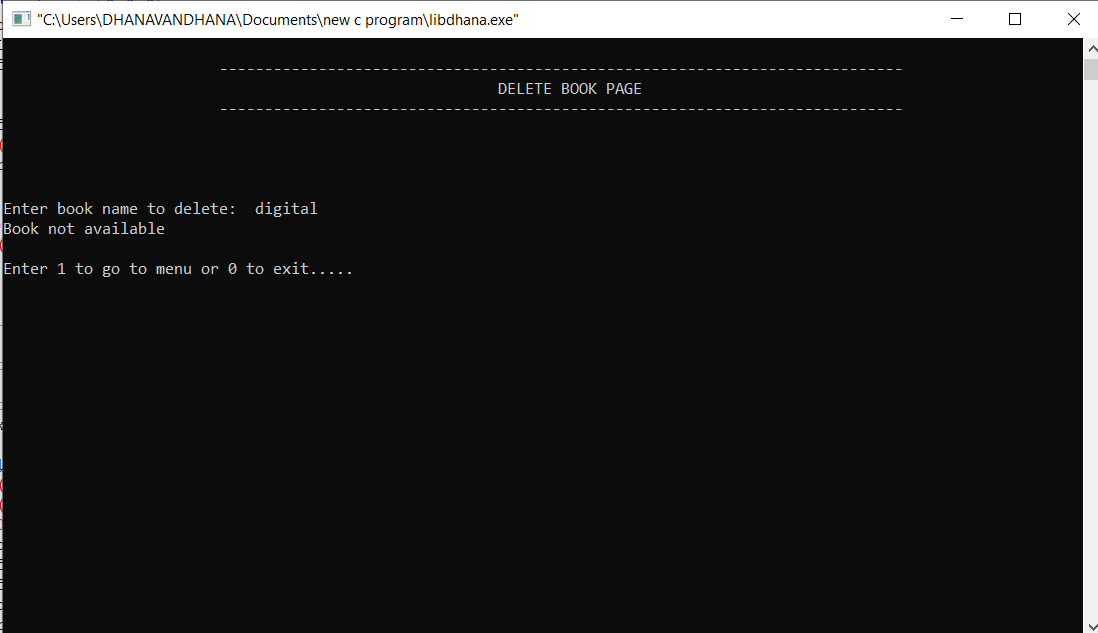
This function is used to delete the books, which are not needed for the library. This function can be accessed by the librarian. This function asks the librarian to enter the book name which he/she wants to delete. Then the book name is compared with all the books available in the library. If the book is available in the library, the book details will be deleted from the library. When the book is deleted, the total count of the book will also decrease. The output screen will display the list of books and total number of books available after deletion. If the book is not available, the function will display a message. Then the books available after deletion will be updated in the file also.

**OUTPUT:**

If book is available:



If book is not available:

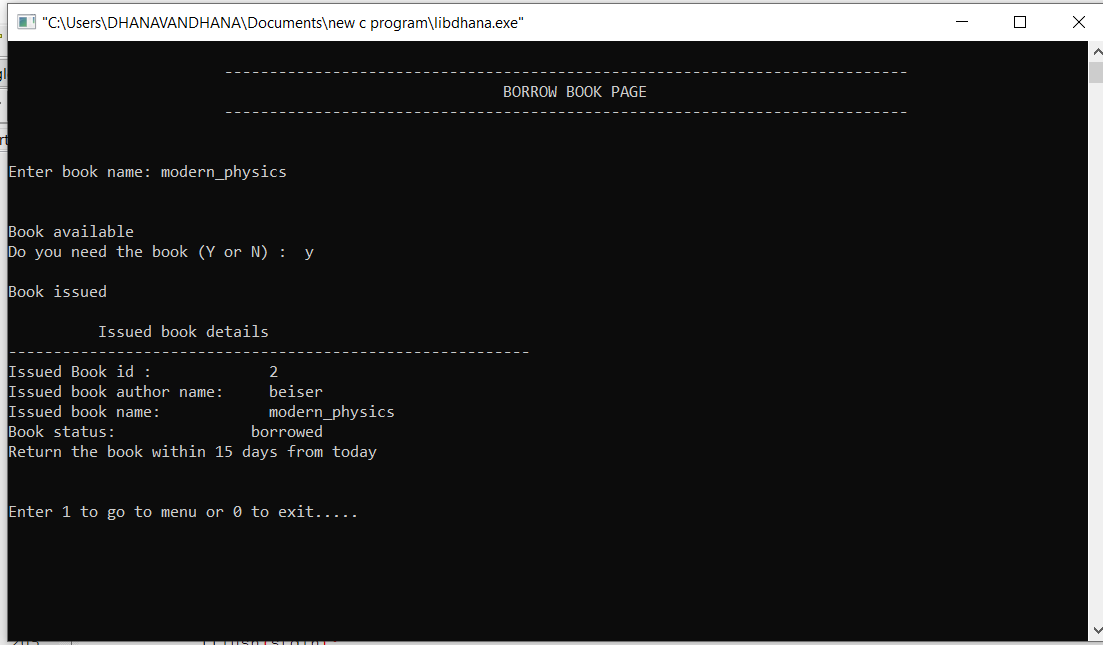


**12. borrow\_book ():**

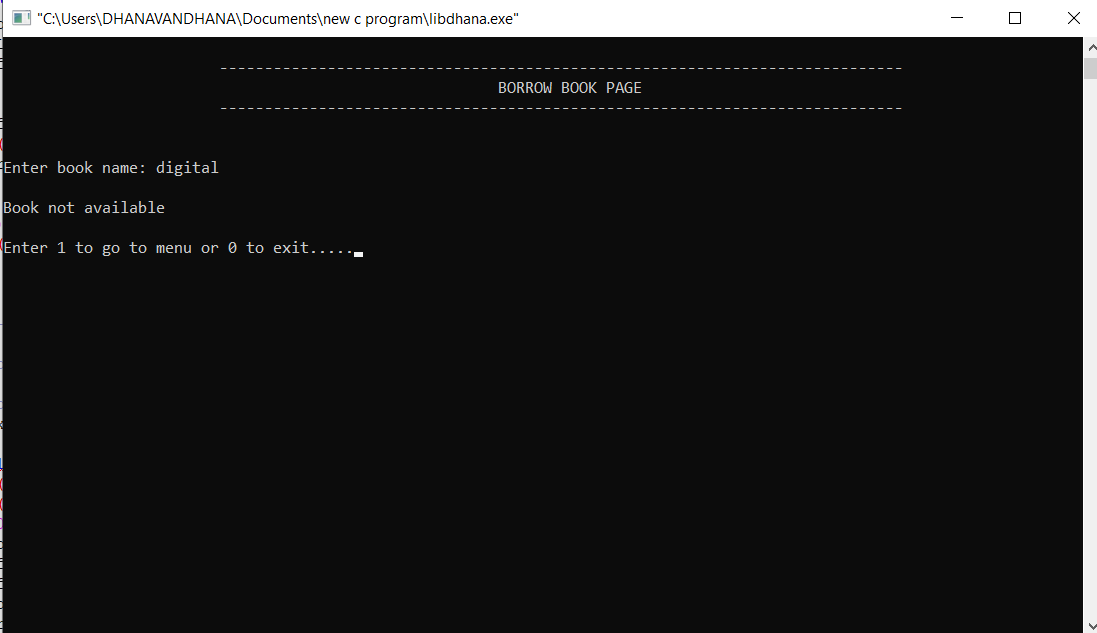
This function is used to borrow books from the library by the students. The student is asked to enter the book name which he/she wants to borrow. The book is checked for its availability in the library. If the book is available and the student needs the book, it will be issued to the student. The output screen will display the borrowed book details. A student is allowed to borrow 5 books only. The number of borrowed books will increase, if the student borrows book from the library. Then the student and book information will be updated in the respective files. If the book is not available, it will display a message**.**

**OUTPUT:**

If book is available:



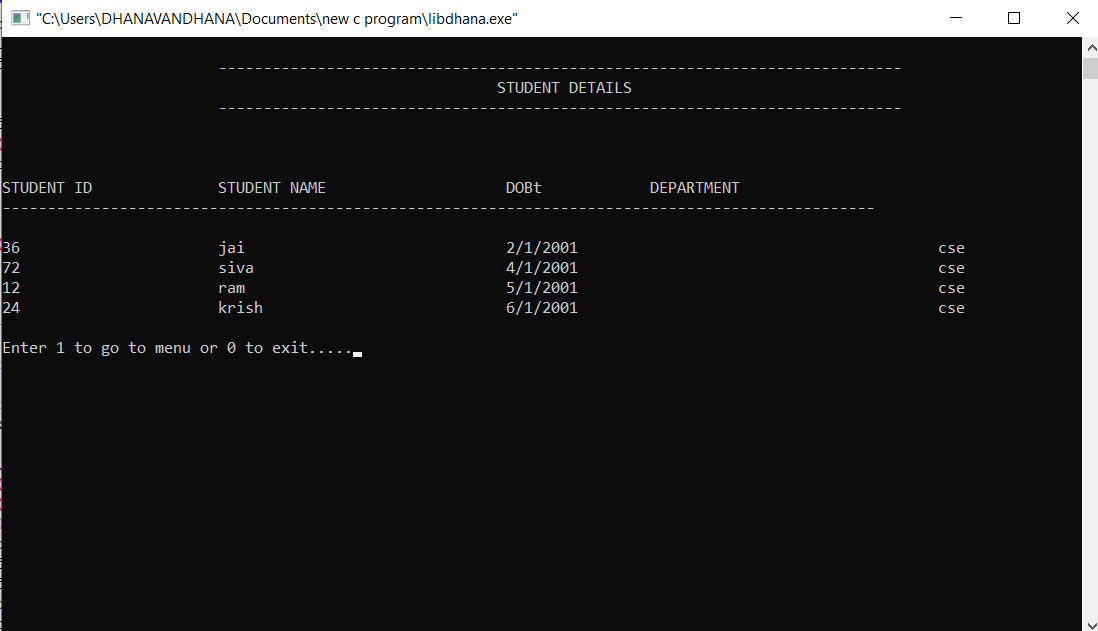
If book is not available:



**13. view\_students ():**

This function will display the list of students who have enrolled in the library. If none of the students have enrolled, it will display a message “No record”.

**OUTPUT:**



**FUNCTIONS USED:**

**1.read\_info ():**

The file containing student information is opened in ‘r’ mode. Data is read from the file and is then stored in the student structure variables. The function also provides the total number of users using the system**.**

**2.update\_info ():**

This function is used to update the student information in the “input.txt” file. When the user edits the student information or borrow or return books, the changes made in the student information will be updated in the file in written mode.

**3.read\_book ():**

This function allows to open the existing file book.txt for reading. Here the library book details are read from the file and stored in an array of book structure and also it gives the total number of books present in the file.

**4.update\_book ():**

This function is to update the status of the books in the library. If the book is borrowed by the student, then it displays ‘borrowed’. If not, it displays ‘available’.

**5.return\_to ():**

This function redirects the user to librarian or student menu, if choice is taken as 1 or else it will exit the page.

**SOURCE CODE:**

#include <stdio.h>

#include <stdlib.h>

#include<string.h>

#define MAX\_BOOK 50

#define MAX\_STUDENT 50

#define USRNAME\_LEN 15

#define PASSWD\_LEN 15

#define NAME\_LEN 15

#define AUTHOR\_LEN 15

const char DEFAULT\_USRNAME[] ="user";

const char DEFAULT\_PASSWRD[] ="password";

int stud\_no;//the position of student info in structure

char user[15];//whether user is student or librarian

int user\_no=0;//total students

int book\_no=0;//total books

//Structure Definitions---------------------------------------------------------------------------------

//date structure

typedef struct

{

int dd,mm,yy;

}Date;

//credentials structure

typedef struct

{

char username[USRNAME\_LEN];

char password[PASSWD\_LEN];

}Check;

//Book structure

typedef struct

{

int id;

char status[15];

char name[NAME\_LEN];

char author[AUTHOR\_LEN];

Date date;

}book;

book b[MAX\_BOOK];

//User structure

typedef struct

{

int id;

char name[15];

char dept[15];

int borrowed\_book\_no;

char borrowed\_books[5][15];

Date dob;

Check check;

}User;

User librarian;

User student[MAX\_STUDENT];

//----------------------------------------------------------------------------------------------------------

int main()

{

first\_page();

return 0;

}

void first\_page()

{

system("cls");

printf("\n\n\n");

printf("\t\t\t----------------------------------------------------------------------\n\n");

printf("\t\t\t| LIBRARY MANAGEMENT SYSTEM |\n\n");

printf("\t\t\t----------------------------------------------------------------------\n\n\n\n\n");

printf("Press any key to continue...");

getch();

enter\_page();

}

//display text as heading

void head\_text(char text[])

{

int len =0;

int pos = 0;

//how many space need to print

len = (78 - strlen(text))/2;

printf("\n\t\t\t----------------------------------------------------------------------------\n");

printf("\t\t\t");

for(pos =0 ; pos < len ; pos++){

printf(" ");

}

printf("%s",text);

printf("\n\t\t\t----------------------------------------------------------------------------\n\n\n");

}

//Login or Sign up

void enter\_page()

{

system("cls");

fflush(stdin);

int choice;

head\_text("ENTER PAGE");

printf("1.Login\n2.User Signup\n\n\nEnter your choice => ");

scanf("%d",&choice);

if(choice==1)

login();

else if(choice==2)

sign\_up();

else{

printf("\n\nINCORRECT CHOICE!!!\nTRY AGAIN!\nPress any key...");

fflush(stdin);

getch();

enter\_page();

}

}

//login as student or librarian

void login()

{

system("cls");

head\_text("LOGIN PAGE");

int choice,loop,trys=0,flag=0;

char username[USRNAME\_LEN];

char password[PASSWD\_LEN];

strcpy((librarian.check.username),DEFAULT\_USRNAME);

strcpy((librarian.check.password),DEFAULT\_PASSWRD);

printf("Enter as \n\n1.STUDENT\n2.LIBRARIAN\n\nEnter choice => ");

scanf("%d",&choice);

read\_info();

if((choice!=1)&&(choice!=2)){

printf("\n\nINCORRECT CHOICE!!!\nTRY AGAIN!\nPress any key...");

fflush(stdin);

getch();

login();

}

else{

do{

strcpy(user,"student");

printf("\nUSERNAME : ");

scanf("%s",&username);

printf("PASSWORD : ");

scanf("%s",&password);

//student login

if(choice==1){

for(loop=0;loop<user\_no;loop++){

if((!strcmp(student[loop].check.username,username))&&(!strcmp(student[loop].check.password,password))){

stud\_no=loop;

flag=1;

break;

}

}

if(flag==1){

student\_menu();

}

}

//librarian login

else if(choice==2){

strcpy(user,"librarian");

if((!strcmp(librarian.check.username,username))&&(!strcmp(librarian.check.password,password))){

librarian\_menu();

break;

}

}

if(flag==0){

printf("\t\t\t\tLogin Failed Enter Again Username & Password\n\n");

trys++;

}

}while(trys<=3);

if(trys>3){

printf("\n\nLogin Failed. Too many trys\n");

printf("\nSorry,Unknown User.");

printf("\nPress any key....");

getch();

first\_page();

}

}

}

//create new student user

void sign\_up()

{

system("cls");

head\_text("SIGN UP PAGE");

read\_info();

int choice,loop,flag=1,id;

char username[USRNAME\_LEN];

char password[PASSWD\_LEN];

enter\_rollno:

printf("ENTER ROLL NO: ");

scanf("%d",&id);

//check if id is unique

for(loop=0;loop<user\_no;loop++){

if(id==student[loop].id){

printf("\n\nROLL NO ALREADY IN USE. Press any key to try again...\n\n");

fflush(stdin);

getch();

goto enter\_rollno;

}

}

enter\_username:

printf("\nUSERNAME: ");

scanf("%s",&username);

//check if username is unique

for(loop=0;loop<user\_no;loop++){

if(!strcmp(username,student[loop].check.username)){

printf("\n\nUSERNAME ALREADY IN USE. Press any key to try again...\n\n");

fflush(stdin);

getch();

goto enter\_username;

}

}

user\_no++;

student[user\_no].id=id;

strcpy(student[user\_no].check.username,username);

printf("\nPASSWORD: ");

scanf("%s",&student[user\_no].check.password);

printf("\nName: ");

scanf("%s",&student[user\_no].name);

printf("\nDepartment: ");

scanf("%s",&student[user\_no].dept);

printf("\nDate Of Birth (dd/mm/yy format): ");

scanf("%d/%d/%d", &student[user\_no].dob.dd, &student[user\_no].dob.mm, &student[user\_no].dob.yy);

student[user\_no].borrowed\_book\_no=0;

strcpy(student[user\_no].borrowed\_books[0],"None\0");

//write to file

FILE \*fp;

fp=fopen("input.txt","a");

fprintf(fp,"\n%d %s %s %s %s %d %d/%d/%d ", student[user\_no].id, student[user\_no].check.username, student[user\_no].check.password, student[user\_no].name, student[user\_no].dept, student[user\_no].borrowed\_book\_no, student[user\_no].dob.dd, student[user\_no].dob.mm, student[user\_no].dob.yy);

fflush(stdin);

fprintf(fp,"%s ",student[user\_no].borrowed\_books[0]);

fflush(stdin);

fclose(fp);

stud\_no=user\_no;

strcpy(user,"student");

student\_menu();

}

//return as specified

void return\_to()

{

int choice;

fflush(stdin);

printf("\n\nEnter 1 to go to menu or 0 to exit.....");

scanf("%d",&choice);

if(choice==1){

if(!strcmp(user,"student"))

student\_menu();

else

librarian\_menu();

}

if(choice==0)

exit(1);

else{

printf("\n\nINCORRECT CHOICE!!!\nTRY AGAIN!\nPress any key...");

return\_to();

}

}

//----------------------------------------------------------------------------------------------------------

//functions to add or update data in file

//read file with user info

void read\_info()

{

int loop,book;

FILE \*fp;

fp=fopen("input.txt","r");

if(fp==NULL){

return;

}

for(loop=0;!(feof(fp));loop++)

{

fscanf(fp,"\n%d %s %s %s %s %d %d/%d/%d ", &student[loop].id, student[loop].check.username, student[loop].check.password, student[loop].name, student[loop].dept, &student[loop].borrowed\_book\_no, &student[loop].dob.dd, &student[loop].dob.mm, &student[loop].dob.yy);

if(student[loop].borrowed\_book\_no==0){

fscanf(fp,"%s ",student[loop].borrowed\_books[0]);

}

else{

for(book=0;book<student[loop].borrowed\_book\_no;book++){

fscanf(fp,"%s ",student[loop].borrowed\_books[book]);

}

}

}

user\_no=loop;

fclose(fp);

}

//read file with book info

void read\_book()

{

int loop;

FILE \*fp;

fp=fopen("books.txt","r");

if(fp==NULL){

fclose(fp);

return;

}

for(loop=0;!(feof(fp));loop++)

fscanf(fp,"\n%d %s %s %s ", &b[loop].id, b[loop].name, b[loop].author, b[loop].status);

book\_no=loop;

fclose(fp);

}

//update changes made in user info to file

void update\_info()

{

int loop,book;

FILE \*fp;

fp=fopen("input.txt","w");

for(loop=0;loop<user\_no;loop++){

fprintf(fp,"\n%d %s %s %s %s %d %d/%d/%d ", student[loop].id, student[loop].check.username, student[loop].check.password, student[loop].name, student[loop].dept, student[loop].borrowed\_book\_no, student[loop].dob.dd, student[loop].dob.mm, student[loop].dob.yy);

fflush(stdin);

if(student[loop].borrowed\_book\_no==0){

fprintf(fp,"%s ","NONE");

fflush(stdin);

}

else{

for(book=0;book<student[loop].borrowed\_book\_no;book++){

fprintf(fp,"%s ",student[loop].borrowed\_books[book]);

fflush(stdin);

}

}

}

fclose(fp);

}

//update changes made in book info to file

void update\_book()

{

FILE \*fp;

fp=fopen("books.txt","w");

int loop=0;

for(loop=0;loop<book\_no;loop++)

{

fprintf(fp,"\n%d %s %s %s ", b[loop].id, b[loop].name, b[loop].author, b[loop].status);

fflush(stdin);

}

fclose(fp);

}

//----------------------------------------------------------------------------------------------------------

void student\_menu()

{

system("cls");

int choice;

head\_text("STUDENT MENU PAGE");

printf("1.View Books\n");

printf("2.Search Books\n");

printf("3.Borrow Book\n");

printf("4.Return Book\n");

printf("5.User Info\n");

printf("6.Edit User Info\n");

printf("7.Add preference\n");

printf("8.Log Out\n");

printf("0.Exit\n");

printf("\n\n\nEnter choice => ");

scanf("%d",&choice);

switch(choice)

{

case 1:

view\_books();

break;

case 2:

search\_book();

break;

case 3:

borrow\_book();

break;

case 4:

return\_book();

break;

case 5:

display\_info();

break;

case 6:

edit\_info();

break;

case 7:

add\_preference();

break;

case 8:

enter\_page();

break;

case 0:

{

printf("\n\n\n\t\t\t\tThank you!!!\n\n\n\n\n");

exit(1);

}

break;

default:

{

printf("\n\n\n\t\t\tINVALID INPUT!!! Try again.Press any key...");

fflush(stdin);

getch();

student\_menu();

}

}

}

void librarian\_menu()

{

system("cls");

int choice;

head\_text("LIBRARIAN MENU PAGE");

printf("1.Add Books\n");

printf("2.Search Books\n");

printf("3.View Books\n");

printf("4.View Students\n");

printf("5.Delete Book\n");

printf("6.Edit User Info\n");

printf("7.View preference\n");

printf("8.Log Out\n");

printf("0.Exit");

printf("\n\n\nEnter choice => ");

scanf("%d",&choice);

switch(choice)

{

case 1:

add\_book();

break;

case 2:

search\_book();

break;

case 3:

view\_books();

break;

case 4:

view\_students();

break;

case 5:

delete\_book();

break;

case 6:

edit\_info();

break;

case 7:

view\_preference();

break;

case 8:

enter\_page();

break;

case 0:

{

printf("\n\n\n\t\t\t\tThank you!!!\n\n\n\n\n");

exit(1);

}

break;

default:

{

printf("\n\n\n\t\t\tINVALID INPUT!!! Try again.Press any key...");

fflush(stdin);

getch();

librarian\_menu();

}

}

}

//----------------------------------------------------------------------------------------------------------

//librarian specific functions

//view students

void view\_students()

{

system("cls");

head\_text("STUDENT DETAILS");

read\_info();

int loop=0;

if(user\_no>0){

printf("\nSTUDENT ID\t\tSTUDENT NAME\t\t\tDOBt\t\tDEPARTMENT\n");

printf("-------------------------------------------------------------------------------------------------\n");

for(loop=0;loop<user\_no;loop++){

printf("\n%d",student[loop].id);

printf("\t\t\t%s",student[loop].name);

printf("\t\t\t\t%d/%d/%d",student[loop].dob.dd,student[loop].dob.mm,student[loop].dob.yy);

printf("\t\t\t\t\t%s",student[loop].dept);

}

}

if(!user\_no)

printf("\n\t\t\tNo Record\n\n");

return\_to();

}

//add book to record

void add\_book()

{

system("cls");

read\_book();

head\_text("ADD BOOKS PAGE");

int loop,id,count=0,flag=1;

FILE \*fp;

fp=fopen("books.txt","a");

printf("\nENTER YOUR DETAILS BELOW:\n\n");

printf("\nBook id: ");

scanf("%d",&id);

for(loop=0;loop<book\_no;loop++){

if(id==b[loop].id){

printf("\n\nBOOK ID not available. Try again. \n\n\nPress any key...");

getch();

flag=0;

break;

}

}

if(flag==0)

add\_book();

book\_no++;

b[book\_no].id=id;

printf("Book name: ");

scanf("%s",&b[book\_no].name);

printf("Author name: ");

scanf("%s",&b[book\_no].author);

strcpy(b[book\_no].status,"available");

fprintf(fp,"%d %s %s %s\n", b[book\_no].id, b[book\_no].name, b[book\_no].author, b[book\_no].status);

fclose(fp);

printf("\n\nBOOK SUCCESSFULLY ADDED\n");

return\_to();

}

//delete book from record

void delete\_book()

{

system("cls");

head\_text("DELETE BOOK PAGE");

read\_book();

int loop,index,count=0;

char name[NAME\_LEN];

printf("\n\nEnter book name to delete: ");

scanf("%s",&name);

for(loop=0;loop<book\_no;loop++){

if(!(strcmp(name,b[loop].name))){

count++;

for(index=loop;index<book\_no;index++){

b[index].id=b[index+1].id;

strcpy(b[index].author,b[index+1].author);

strcpy(b[index].name,b[index+1].name);

strcpy(b[index].status,b[index+1].status);

}

book\_no--;

--loop;

}

}

if(count==0)

printf("Book not available");

else{

printf("\n Books available after deletion ");

printf("\n---------------------------------------------------------\n");

printf("BOOK ID\t\tAUTHOR\t\t\tNAME\t\t\t\tSTATUS");

for(loop=0;loop<book\_no;loop++){

printf("\n%d",b[loop].id);

printf("\t\t%s",b[loop].author);

printf("\t\t\t%s",b[loop].name);

printf("\t\t\t\t%s",b[loop].status);

}

printf("\n\n\nTotal number of books available after deletion: %d",book\_no);

printf("\n\n\n\n");

}

update\_book();

return\_to();

}

void view\_preference()

{

system("cls");

head\_text("VIEW PREFERENCES");

FILE \*fp;

fp=fopen("prefer.txt","r");

char author[AUTHOR\_LEN],name[NAME\_LEN];

if(fp==NULL){

printf("\t\tNo record of preferences!!\n\n");

}

else{

printf("\t\t\tBOOK NAMES\t\tAUTHOR NAMES\n\n");

while((fscanf(fp,"%s %s\n",name,author))!=EOF){

printf("\t\t\t%s\t - \t%s\n",name,author);

}

}

fclose(fp);

return\_to();

}

//----------------------------------------------------------------------------------------------------------

//functions common to both student and librarian

//list all books in the record

void view\_books()

{

system("cls");

head\_text("BOOK DETAILS");

read\_book();

int loop=0;

if(book\_no>0){

printf("\nBOOK ID\t\t\tBOOK NAME\t\t\tAUTHOR NAME\t\t\tBOOK STATUS\n");

printf("-------------------------------------------------------------------------------------------------\n");

for(loop=0;loop<book\_no;loop++){

printf("\n%d",b[loop].id);

printf("\t\t\t%s",b[loop].name);

printf("\t\t\t\t%s",b[loop].author);

printf("\t\t\t\t\t%s",b[loop].status);

}

}

if(!book\_no)

printf("\n\t\t\tNo Record\n\n");

return\_to();

}

//display user info

void display\_info()

{

system("cls");

int i;

head\_text("STUDENT INFO");

read\_info();

printf("STUDENT ID: %d\n",student[stud\_no].id);

printf("NAME : %s\n",student[stud\_no].name);

printf("DEPARTMENT : %s\n",student[stud\_no].dept);

printf("DOB: %d/%d/%d\n", student[stud\_no].dob.dd, student[stud\_no].dob.mm, student[stud\_no].dob.yy);

printf("NO OF BOOKS BORROWED : %d\n",student[stud\_no].borrowed\_book\_no);

if(student[stud\_no].borrowed\_book\_no>0){

printf("BORROWED BOOKS : \n\n");

for(i=0;i<student[stud\_no].borrowed\_book\_no;i++){

printf("%s\n",student[stud\_no].borrowed\_books[i]);

}

}

return\_to();

}

//edit student info

void edit\_info()

{

system("cls");

head\_text("EDIT INFO");

read\_info();

int loop,id,flag=0;

char username[15];

if(!strcmp(user,"student"))

strcpy(username,student[stud\_no].check.username);

else{

printf("Enter username: ");

scanf("%s",username);

for(loop=0;loop<user\_no;loop++){

if(!(strcmp(username,student[loop].check.username))){

flag=1;

stud\_no=loop;

break;

}

}

if(flag==0){

printf("USERNAME not found. Try again. Press any key...");

getch();

fflush(stdin);

edit\_info();

}

}

printf("\nCURRENT INFO\n\n");

printf("STUDENT ID: %d\n",student[stud\_no].id);

printf("USERNAME : %s\n",student[stud\_no].check.username);

printf("NAME : %s\n",student[stud\_no].name);

printf("DEPARTMENT : %s\n",student[stud\_no].dept);

printf("DATE OF BIRTH: %d/%d/%d\n", student[stud\_no].dob.dd, student[stud\_no].dob.mm, student[stud\_no].dob.yy);

printf("\n\nEnter modified Info\n\n");

check\_id:

printf("STUDENT ID: ");

scanf("%d",&id);

//check if unique

for(loop=0;loop<user\_no;loop++){

if((student[loop].id==id)&&(stud\_no!=loop)){

printf("ID is not unique. TRY again.\n\n");

fflush(stdin);

goto check\_id;

}

}

check\_username:

printf("USERNAME : ");

scanf("%s",username);

for(loop=0;loop<user\_no;loop++){

if((!(strcmp(username,student[loop].check.username))&&(stud\_no!=loop))){

printf("USERNAME is not unique. TRY again.\n\n");

fflush(stdin);

goto check\_username;

}

}

student[stud\_no].id=id;

strcpy(student[stud\_no].check.username,username);

printf("NAME : ");

scanf("%s",student[stud\_no].name);

printf("DEPARTMENT : ");

scanf("%s",student[stud\_no].dept);

printf("DATE OF BIRTH : ");

scanf("%d/%d/%d", &student[stud\_no].dob.dd, &student[stud\_no].dob.mm, &student[stud\_no].dob.yy);

update\_info();

return\_to();

}

void search\_book()

{

system("cls");

head\_text("SEARCH BOOK");

read\_book();

int choice,loop,flag=0,avail\_book=0;

char author[AUTHOR\_LEN],name[NAME\_LEN];

printf("Search by : \n\n");

printf("1.Title\n2.Author\n\n\nEnter choice => ");

scanf("%d",&choice);

if(choice==1){

printf("\n\nEnter the Title of the book : ");

scanf("%s",&name);

for(loop=0;loop<book\_no;loop++){

if(strcmp(name,b[loop].name)==0){

avail\_book++;

if(avail\_book==1){

printf("\nBOOK IS AVAILABLE\n\n\t\t\tBOOK DETAILS\n\n");

}

printf("BOOK NO %d : \n",avail\_book);

printf("BOOK ID :\t%d\n",b[loop].id);

printf("BOOK NAME :\t%s\n",b[loop].name);

printf("AUTHOR NAME :\t%s\n",b[loop].author);

printf("BOOK STATUS :\t%s\n",b[loop].status);

printf("---------------------------------------------------\n\n");

flag=1;

continue;

}

}

}

else if(choice==2){

printf("\n\nEnter Author name : ");

scanf("%s",&author);

for(loop=0;loop<book\_no;loop++){

if(strcmp(author,b[loop].author)==0){

avail\_book++;

if(avail\_book==1){

printf("\nBOOK IS AVAILABLE\n\n\t\t\tBOOK DETAILS\n\n");

}

printf("BOOK NO %d : \n",avail\_book);

printf("BOOK ID :\t%d\n",b[loop].id);

printf("BOOK NAME :\t%s\n",b[loop].name);

printf("AUTHOR NAME :\t%s\n",b[loop].author);

printf("BOOK STATUS :\t%s\n",b[loop].status);

printf("PUBLISHED DATE :\t%d/%d/%d\n",b[loop].date.dd,b[loop].date.mm,b[loop].date.yy);

printf("---------------------------------------------------\n\n");

flag=1;

continue;

}

}

}

else{

printf("\n\n\nWRONG CHOICE!!\nTRY AGAIN.\n...");

fflush(stdin);

getch();

search\_book();

}

if(flag==0)

printf("\n\n\nWE DON'T HAVE THE BOOK YOU REQUIRED\n");

return\_to();

}

//----------------------------------------------------------------------------------------------------------

//student specific functions

//return book

void return\_book()

{

system("cls");

head\_text("RETURN BOOK PAGE");

read\_book();

int loop,flag=0,book,temp,num,pos;

char name[NAME\_LEN],status[15];

printf("Enter book name: ");

scanf("%s",&name);

for(loop=0;loop<student[stud\_no].borrowed\_book\_no;loop++)

{

if(!(strcmp(name,student[stud\_no].borrowed\_books[loop])))

{

//find position of book in structure

for(book=0;book<book\_no;book++){

if(!(strcmp(b[book].name,name))){

pos=book;

}

}

flag=1;

printf("\nBook issued");

printf("\n\n Returned book details ");

printf("\n----------------------------------------------------------");

printf("\nReturned Book id : %d",b[pos].id);

printf("\nReturned book author name: %s",b[pos].author);

printf("\nReturned book name: %s",b[pos].name);

strcpy(b[loop].status,"available");

printf("\nBook status: %s",b[pos].status);

num=student[stud\_no].borrowed\_book\_no;

//remove from borrowed books

for(book=0;book<num;book++){

if(!(strcmp(student[stud\_no].borrowed\_books[book],name))){

for(temp=book;temp<num;temp++){

strcpy(student[stud\_no].borrowed\_books[temp],student[stud\_no].borrowed\_books[temp+1]);

}

break;

}

}

student[stud\_no].borrowed\_book\_no--;

update\_info();

update\_book();

break;

}

else

continue;

}

if(flag==0)

printf("\nBook not found to be borrowed! ");

return\_to();

}

//borrow book

void borrow\_book()

{

system("cls");

head\_text("BORROW BOOK PAGE");

read\_book();

int loop,flag=0,num;

char c,name[NAME\_LEN],status[15];

printf("Enter book name: ");

scanf("%s",&name);

for(loop=0;loop<book\_no;loop++)

{

if((!(strcmp(name,b[loop].name)))&&(!(strcmp(b[loop].status,"available"))))

{

flag=1;

printf("\n\nBook available");

printf("\nDo you need the book (Y or N) : ");

fflush(stdin);

scanf("%c",&c);

if((c=='y')||(c=='Y')){

if(student[stud\_no].borrowed\_book\_no==5){

printf("\n\nA USER CANNOT BORROW MORE THAN 5 BOOKS\n\n");

return\_to();

}

printf("\nBook issued\n");

printf("\n Issued book details ");

printf("\n----------------------------------------------------------");

printf("\nIssued Book id : %d",b[loop].id);

printf("\nIssued book author name: %s",b[loop].author);

printf("\nIssued book name: %s",b[loop].name);

strcpy(b[loop].status,"borrowed");

printf("\nBook status: %s",b[loop].status);

num=student[stud\_no].borrowed\_book\_no;

strcpy(student[stud\_no].borrowed\_books[num],b[loop].name);

student[stud\_no].borrowed\_book\_no++;

update\_info();

update\_book();

printf("\nReturn the book within 15 days from today\n");

break;

}

}

}

if(flag==0)

printf("\nBook not available");

return\_to();

}

void add\_preference()

{

system("cls");

head\_text("REQUEST TO ADD PREFERRED BOOKS IN THE LIBRARY");

read\_book();

int loop,flag=0;

char name[NAME\_LEN],author[AUTHOR\_LEN];

printf("Enter the title of the book : ");

scanf("%s",name);

printf("Enter the author of the book : ");

scanf("%s",author);

for(loop=0;loop<book\_no;loop++){

if((!strcasecmp(b[loop].name,name))&&(!strcmp(b[loop].author,author))){

flag=1;

printf("\n\t\t\tTHE BOOK YOU PREFERED IS ALREADY AVAILABLE IN THE LIBRARY!! CHECK IN VIEW BOOKS!!\n\n");

break;

}

}

if(flag==0){

FILE \*fp;

fp=fopen("prefer.txt","a");

fprintf(fp,"%s %s\n",name,author);

fflush(stdin);

fclose(fp);

printf("\n\t\tYOUR PREFERENCES ARE NOTED!! WE WILL TRY TO ADD THE BOOK AS SOON AS POSSIBLE!!\n\n");

}

return\_to();

}

**CONCLUSION:**

After we have completed the project, we are sure that our system will overcome certain problems in the existing system. The “LIBRARY MANAGEMENT SYSTEM” process is computerized to reduce human errors and to increase the efficiency. The main focus of this project is to lessen human efforts. The maintenance of the records is made efficient and can be retrieved easily. This simple system can be used in a library to insert, store, handle and retrieve information about books. We hope that this project will make the use of library efficient, user friendly, easier and effective. And it is expected that this project will go a long way in satisfying users’ requirements. The computerization of the Library Management will not only improve the efficiency but will also reduce human stress.

**FUTURE WORK:**

User requirements keep changing as the system is being used. Future releases should be made to cope with the changing requirements. Updates should be recognized and accepted. The updates can contain the following features:

* A billing system for collection of fines.
* E-books access through links. We can attach the links of the e-books in the code, so that the users can access the e-books whenever they are in need for the book.
* A notification system to can be added to notify users of important announcements, due date of books, fines and current changes in the system.
* Graphical User interface can be added for more user-friendly environment.
* Advanced encryption techniques for improved security.

**END**