LIBRARY MANAGEMENT SYSTEM IN C

```c
#include <stdio.h>
#include <time.h>
#include <string.h>
#define MAX_YR 9999
#define MIN_YR 1900
#define MAX_SIZE_USER_NAME 30
#define MAX_SIZE_PASSWORD 20
#define FILE_NAME "AticleworldLibBookS.bin"
// Macro related to the books info
#define MAX_BOOK_NAME 50
#define MAX_AUTHOR_NAME 50
#define MAX_STUDENT_NAME 50
#define MAX_STUDENT_ADDRESS 300
#define FILE_HEADER_SIZE sizeof(sFileHeader)
//structure to store date
typedef struct
{
int yyyy;
int mm;
int dd;
} Date;
typedef struct
{
char username[MAX_SIZE_USER_NAME];
char password[MAX_SIZE_PASSWORD];
} sFileHeader;
typedef struct// to call in program
{
unsigned int books_id; // declare the integer data type
char bookName[MAX_BOOK_NAME];// declare the character data type
char authorName[MAX_AUTHOR_NAME];// declare the charecter data type
char studentName[MAX_STUDENT_NAME];// declare the character data type
char studentAddr[MAX_STUDENT_ADDRESS];// declare the character data type
Date bookIssueDate;// declare the integer data type
} s_BooksInfo;
void printMessageCenter(const char* message)
{
int len =0;
int pos = 0;
//calculate how many space need to print
len = (78 - strlen(message))/2;
printf("\t\t\t");
for(pos =0 ; pos < len ; pos++)
{
```

```
//print space
printf(" ");
}
//print message
printf("%s",message);
}
void headMessage(const char *message)
{
system("cls");
printf("\t\t\t###############################################################################");
printf("\n\t\t\t############# #############");
printf("\n\t\t\t############# Library management System Project in C #############");
printf("\n\t\t\t############# #############");
printf("\n\t\t\t###############################################################################");
printf("\n\t\t\t--------------------------------------------------------------------------\n");
printMessageCenter(message);
printf("\n\t\t\t--------------------------------------------------------------------------");
}
void welcomeMessage()
{
headMessage("www.aticleworld.com");
printf("\n\n\n\n\n");
printf("\n\t\t\t **-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**\n");
printf("\n\t\t\t =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=");
printf("\n\t\t\t = WELCOME =");
printf("\n\t\t\t = TO =");
printf("\n\t\t\t = LIBRARY =");
printf("\n\t\t\t = MANAGEMENT =");
printf("\n\t\t\t = SYSTEM =");
printf("\n\t\t\t =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=");
printf("\n\t\t\t **-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**\n");
printf("\n\n\n\t\t\t Enter any key to continue.....");
getch();
}
int isNameValid(const char *name)
{
int validName = 1;
int len = 0;
int index = 0;
len = strlen(name);
for(index =0; index <len ; ++index)
{
if(!(isalpha(name[index])) && (name[index] != '\n') && (name[index] != ' '))
{
validName = 0;
```

```
break;
}
}
return validName;
}
// Function to check leap year.
//Function returns 1 if leap year
int IsLeapYear(int year)
{
return (((year % 4 == 0) &&
(year % 100 != 0)) ||
(year % 400 == 0));
}
// returns 1 if given date is valid.
int isValidDate(Date *validDate)
{
//check range of year,month and day
if (validDate->yyyy > MAX_YR ||
validDate->yyyy < MIN_YR)
return 0;
if (validDate->mm < 1 || validDate->mm > 12)
return 0;
if (validDate->dd < 1 || validDate->dd > 31)
return 0;
//Handle feb days in leap year
if (validDate->mm == 2)
{
if (IsLeapYear(validDate->yyyy))
return (validDate->dd <= 29);
else
return (validDate->dd <= 28);
}
//handle months which has only 30 days
if (validDate->mm == 4 || validDate->mm == 6 ||
validDate->mm == 9 || validDate->mm == 11)
return (validDate->dd <= 30);
return 1;
}
// Add books in list
void addBookInDataBase()
{
int days;
s_BooksInfo addBookInfoInDataBase = {0};
FILE *fp = NULL;
int status = 0;
```

```
fp = fopen(FILE_NAME,"ab+");
if(fp == NULL)
{
printf("File is not opened\n");
exit(1);
}
headMessage("ADD NEW BOOKS");
printf("\n\n\t\t\tENTER YOUR DETAILS BELOW:");
printf("\n\t\t\t-------------------------------------------------------------------------\n");
printf("\n\t\t\tBook ID NO = ");
fflush(stdin);
scanf("%u",&addBookInfoInDataBase.books_id);
do
{
printf("\n\t\t\tBook Name = ");
fflush(stdin);
fgets(addBookInfoInDataBase.bookName,MAX_BOOK_NAME,stdin);
status = isNameValid(addBookInfoInDataBase.bookName);
if (!status)
{
printf("\n\t\t\tName contain invalid character. Please enter again.");
}
}
while(!status);
do
{
printf("\n\t\t\tAuthor Name = ");
fflush(stdin);
fgets(addBookInfoInDataBase.authorName,MAX_AUTHOR_NAME,stdin);
status = isNameValid(addBookInfoInDataBase.authorName);
if (!status)
{
printf("\n\t\t\tName contain invalid character. Please enter again.");
}
}
while(!status);
do
{
printf("\n\t\t\tStudent Name = ");
fflush(stdin);
fgets(addBookInfoInDataBase.studentName,MAX_STUDENT_NAME,stdin);
status = isNameValid(addBookInfoInDataBase.studentName);
if (!status)
{
printf("\n\t\t\tName contain invalid character. Please enter again.");
```

```
}
}
while(!status);
do
{
//get date year,month and day from user
printf("\n\t\t\tEnter date in format (day/month/year): ");
scanf("%d/%d/%d",&addBookInfoInDataBase.bookIssueDate.dd,&addBookInfoInDataBase.bookIssueDate.mm,&addBookInfoInDataBase.bookIss
ueDate.yyyy);
//check date validity
status = isValidDate(&addBookInfoInDataBase.bookIssueDate);
if (!status)
{
printf("\n\t\t\tPlease enter a valid date.\n");
}
}
while(!status);
fwrite(&addBookInfoInDataBase,sizeof(addBookInfoInDataBase), 1, fp);
fclose(fp);
}
// search books
void searchBooks()
{
int found = 0;
char bookName[MAX_BOOK_NAME] = {0};
s_BooksInfo addBookInfoInDataBase = {0};
FILE *fp = NULL;
int status = 0;
fp = fopen(FILE_NAME,"rb");
if(fp == NULL)
{
printf("\n\t\t\tFile is not opened\n");
exit(1);
}
headMessage("SEARCH BOOKS");
//put the control on books detail
if (fseek(fp,FILE_HEADER_SIZE,SEEK_SET) != 0)
{
fclose(fp);
printf("\n\t\t\tFacing issue while reading file\n");
exit(1);
}
printf("\n\n\t\t\tEnter Book Name to search:");
fflush(stdin);
fgets(bookName,MAX_BOOK_NAME,stdin);
```

```
while (fread (&addBookInfoInDataBase, sizeof(addBookInfoInDataBase), 1, fp))
{
if(!strcmp(addBookInfoInDataBase.bookName, bookName))
{
found = 1;
break;
}
}
if(found)
{
printf("\n\t\t\tBook id = %u\n",addBookInfoInDataBase.books_id);
printf("\t\t\tBook name = %s",addBookInfoInDataBase.bookName);
printf("\t\t\tBook authorName = %s",addBookInfoInDataBase.authorName);
printf("\t\t\tBook issue date(day/month/year) = (%d/%d/%d)",addBookInfoInDataBase.bookIssueDate.dd,
addBookInfoInDataBase.bookIssueDate.mm, addBookInfoInDataBase.bookIssueDate.yyyy);
}
else
{
printf("\n\t\t\tNo Record");
}
fclose(fp);
printf("\n\n\n\t\t\tPress any key to go to main menu.....");
getchar();
}
// v books function
void viewBooks()
{
int found = 0;
char bookName[MAX_BOOK_NAME] = {0};
s_BooksInfo addBookInfoInDataBase = {0};
FILE *fp = NULL;
int status = 0;
unsigned int countBook = 1;
headMessage("VIEW BOOKS DETAILS");
fp = fopen(FILE_NAME,"rb");
if(fp == NULL)
{
printf("File is not opened\n");
exit(1);
}
if (fseek(fp,FILE_HEADER_SIZE,SEEK_SET) != 0)
{
fclose(fp);
printf("Facing issue while reading file\n");
exit(1);
```

```
}
while (fread (&addBookInfoInDataBase, sizeof(addBookInfoInDataBase), 1, fp))
{
printf("\n\t\t\tBook Count = %d\n\n",countBook);
printf("\t\t\tBook id = %u",addBookInfoInDataBase.books_id);
printf("\n\t\t\tBook name = %s",addBookInfoInDataBase.bookName);
printf("\t\t\tBook authorName = %s",addBookInfoInDataBase.authorName);
printf("\t\t\tBook issue date(day/month/year) = (%d/%d/%d)\n\n",addBookInfoInDataBase.bookIssueDate.dd,
addBookInfoInDataBase.bookIssueDate.mm, addBookInfoInDataBase.bookIssueDate.yyyy);
found = 1;
++countBook;
}
fclose(fp);
if(!found)
{
printf("\n\t\t\tNo Record");
}
printf("\n\n\t\t\tPress any key to go to main menu.....");
fflush(stdin);
getchar();
}
// delete function
void deleteBooks()
{
int found = 0;
int bookDelete = 0;
sFileHeader fileHeaderInfo = {0};
char bookName[MAX_BOOK_NAME] = {0};
s_BooksInfo addBookInfoInDataBase = {0};
FILE *fp = NULL;
FILE *tmpFp = NULL;
int status = 0;
headMessage("Delete Books Details");
fp = fopen(FILE_NAME,"rb");
if(fp == NULL)
{
printf("File is not opened\n");
exit(1);
}
tmpFp = fopen("tmp.bin","wb");
if(tmpFp == NULL)
{
fclose(fp);
printf("File is not opened\n");
exit(1);
```

```
}
fread (&fileHeaderInfo,FILE_HEADER_SIZE, 1, fp);
fwrite(&fileHeaderInfo,FILE_HEADER_SIZE, 1, tmpFp);
printf("\n\t\t\tEnter Book ID NO. for delete:");
scanf("%d",&bookDelete);
while (fread (&addBookInfoInDataBase, sizeof(addBookInfoInDataBase), 1, fp))
{
if(addBookInfoInDataBase.books_id != bookDelete)
{
fwrite(&addBookInfoInDataBase,sizeof(addBookInfoInDataBase), 1, tmpFp);
}
else
{
found = 1;
}
}
(found)? printf("\n\t\t\tRecord deleted successfully....."):printf("\n\t\t\tRecord not found");
fclose(fp);
fclose(tmpFp);
remove(FILE_NAME);
rename("tmp.bin",FILE_NAME);
}
void updateCredential(void)
{
sFileHeader fileHeaderInfo = {0};
FILE *fp = NULL;
unsigned char userName[MAX_SIZE_USER_NAME] = {0};
unsigned char password[MAX_SIZE_PASSWORD] = {0};
headMessage("Update Credential");
fp = fopen(FILE_NAME,"rb+");
if(fp == NULL)
{
printf("File is not opened\n");
exit(1);
}
fread (&fileHeaderInfo,FILE_HEADER_SIZE, 1, fp);
if (fseek(fp,0,SEEK_SET) != 0)
{
fclose(fp);
printf("\n\t\t\tFacing issue while updating password\n");
exit(1);
}
printf("\n\n\t\t\tNew Username:");
fflush(stdin);
fgets(userName,MAX_SIZE_USER_NAME,stdin);
```

```
printf("\n\n\t\t\tNew Password:");
fflush(stdin);
fgets(password,MAX_SIZE_PASSWORD,stdin);
strncpy(fileHeaderInfo.username,userName,sizeof(userName));
strncpy(fileHeaderInfo.password,password,sizeof(password));
fwrite(&fileHeaderInfo,FILE_HEADER_SIZE, 1, fp);
fclose(fp);
printf("\n\t\t\tYour Password has been changed successfully");
printf("\n\t\t\ttLogin Again:");
fflush(stdin);
getchar();
exit(1);
}
void menu()
{
int choice = 0;
do
{
headMessage("MAIN MENU");
printf("\n\n\n\t\t\t1.Add Books");
printf("\n\t\t\t2.Search Books");
printf("\n\t\t\t3.View Books");
printf("\n\t\t\t4.Delete Book");
printf("\n\t\t\t5.Update Password");
printf("\n\t\t\t0.Exit");
printf("\n\n\n\t\t\tEnter choice => ");
scanf("%d",&choice);
switch(choice)
{
case 1:
addBookInDataBase();
break;
case 2:
searchBooks();
break;
case 3:
viewBooks();
break;
case 4:
deleteBooks();
break;
case 5:
updateCredential();
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...");
} //Switch Ended
}
while(choice!=0); //Loop Ended
}
//login password
void login()
{
unsigned char userName[MAX_SIZE_USER_NAME] = {0};
unsigned char password[MAX_SIZE_PASSWORD] = {0};
int L=0;
sFileHeader fileHeaderInfo = {0};
FILE *fp = NULL;
headMessage("Login");
fp = fopen(FILE_NAME,"rb");
if(fp == NULL)
{
printf("File is not opened\n");
exit(1);
}
fread (&fileHeaderInfo,FILE_HEADER_SIZE, 1, fp);
fclose(fp);
do
{
printf("\n\n\n\t\t\t\tUsername:");
fgets(userName,MAX_SIZE_USER_NAME,stdin);
printf("\n\t\t\t\tPassword:");
fgets(password,MAX_SIZE_PASSWORD,stdin);
if((!strcmp(userName,fileHeaderInfo.username)) && (!strcmp(password,fileHeaderInfo.password)))
{
menu();
}
else
{
printf("\t\t\t\tLogin Failed Enter Again Username & Password\n\n");
L++;
}
}
while(L<=3);
if(L>3)
{
```

```
headMessage("Login Failed");
printf("\t\t\t\tSorry,Unknown User.");
getch();
system("cls");
}
}
int isFileExists(const char *path)
{
// Try to open file
FILE *fp = fopen(path, "rb");
int status = 0;
// If file does not exists
if (fp != NULL)
{
status = 1;
// File exists hence close file
fclose(fp);
}
return status;
}
void init()
{
FILE *fp = NULL;
int status = 0;
const char defaultUsername[] ="aticleworld\n";
const char defaultPassword[] ="aticleworld\n";
sFileHeader fileHeaderInfo = {0};
status = isFileExists(FILE_NAME);
if(!status)
{
//create the binary file
fp = fopen(FILE_NAME,"wb");
if(fp != NULL)
{
//Copy default password
strncpy(fileHeaderInfo.password,defaultPassword,sizeof(defaultPassword));
strncpy(fileHeaderInfo.username,defaultUsername,sizeof(defaultUsername));
fwrite(&fileHeaderInfo,FILE_HEADER_SIZE, 1, fp);
fclose(fp);
}
}
}
int main()
{
init();
```

```
welcomeMessage();
login();
return 0;
}
```