Library Management System

Intro to C Programming

CSCI - 1110

MEDKS

University of New Haven
TAGLIATELA COLLEGE OF ENGINEERING, West Haven, CT

Submitted To:
Dr. Reza Sadeghi

Fall 2020

Final Project Report of Library Management System

Team Name

MEDKS

Members

Matthew Lamour	mlamo1@unh.newhaven.edu (Team Lead)
Eon Sattaur	esatt1@unh.newhaven.edu (Team Member)
Daniella Weber	dwebe2@unh.newhaven.edu (Team Member)
Thomas Knecht	tknec1@unh.newhaven.edu (Team Member)
Steven LeClerc	slecl1@unh.newhaven.edu (Team Member)
Kyle Muldoon	kmuld1@unh.newhaven.edu (Team Member)

Roles of Team Members

Matthew Lamour

Tasked with creating administration user capabilities:

- Required LMS admin username and password log-in
- Change administrator username and password when desired
- Add guest users to the LMS by creating usernames and passwords
    - Note: guest users do not have the same permissions as the admin user
- Remove guest users by removing all guest associated usernames, passwords, and corresponding recorded files
- Add item(s) to the LMS with varied details
- Delete / edit item(s) in the LMS
- View recorded list of borrowing requests
- Accept / reject borrowing request(s)

Tasked with implementing the encrypt/decrypt functions to protect user information.
Tasked with writing the formal report document sections: Variables and Functions.

Eon Sattaur

Tasked with creating LMS user-friendly features:

- A welcome page
- A menu of all functions available to the user / admin
- Illustrated reports of requested / returned item(s) in tabular form
- LMS exit function with a ‘Thank You’ message
- Warning labels if the administrator tries to add a new, pre-existing item to the inventory list, a guest user tries to borrow more than three items at a time, or a user search request return null items

Daniella Weber

Tasked with creating LMS user-friendly features:

- A welcome page
- A menu of all functions available to the user / admin
- Illustrated reports of requested / returned item(s) in tabular form
- LMS exit function with a 'Thank You' message
- Warning labels if the administrator tries to add a new, pre-existing item to the inventory list, a guest user tries to borrow more than three items at a time, or a user search request return null items

Tasked with creating the original LMS inventory list.

Tasked with writing the formal report document.

Tasked with creating the formal report presentation (Powerpoint).

Thomas Knecht

Tasked with creating LMS user capabilities:

- Search through LMS based on all item details
- Save a list of favorite books
- Request for borrowing of item(s) for a specific time
- View history of borrowed items

Tasked with creating the original LMS inventory list.

Steven LeClerc

Tasked with creating LMS user capabilities:

- Search through LMS based on all item details
- Save a list of favorite books
- Request for borrowing of item(s) for a specific time
- View history of borrowed items

Tasked with creating the original LMS inventory list.

Kyle Muldoon

Tasked with creating administration user capabilities:

- Required LMS admin username and password log-in
- Change administrator username and password when desired
- Add guest users to the LMS by creating usernames and passwords
    - Note: guest users do not have the same permissions as the admin user
- Remove guest users by removing all guest associated usernames, passwords, and corresponding recorded files
- Add item(s) to the LMS with varied details
- Delete / edit item(s) in the LMS
- View recorded list of borrowing requests
- Accept / reject borrowing request(s)

Table of Contents

Table of Lists

Test Library Inventory

Inventory Format: ID, Title, Author, Subject, ISBN, Format, Location, Quantity, Price

01, Atlas, The Score, Music, 6739820817, 12, Shelf 888, $9.99
02, Harry Potter, J.K Rowling, Literature, 2745394572, 8, Shelf 777, $11.00
03, Animal Farm, George Orwell, Literature, 9780241196, 7, Shelf 321, $5.99
04, Fahrenheit 451, Ray Bradbury, Literature, 8821407053, 1, Shelf 101, $14.99
05, Calculus for Dummies, Mark Ryan, Literature, 9780764524, 17, Shelf 54, $17.99
06, Art of War, Sun Tzu, Literature, 9781982530, 9, Shelf 985, $3.99
07, Cleopatra, The Lumineers, Music, 225830971, 15, Shelf 762, $7.99
08, Weapons of Math Destruction, Cathy O’Neil, Literature, 4782938871, 3, Shelf 66, $4.99
09, Die Hard Series, 20th Century Studios, Film, 8938094712, 5, Shelf 151, $24.99
10, A Night at the Opera, Queen, Music, 3303981343, 12, Shelf 413, $7.99

Test User Data

User Format: Name, Password, Type, ID [Books Loans and Requests as well, but not in table]

testname1, testpass1, ADMIN, 1
testname2, testpass2, GUEST, 2
testname3, testpass3, GUEST, 3
testname4, testpass4, GUEST, 4

Table of Text Files

List of Users

A separate text file, “lmsusers.txt” that lists user information: administrator and all possible guests, as well as borrow requests associated with specific users. This file is decrypted upon it being read by the LMS software and is encrypted upon closing the LMS software.

If this file does not exist, it will be created and filled with test data from the above Test User Data Table.

List of Library Items

A separate text file, “lmslibrary.txt” that lists the library inventory: all items and associated details.

If this file does not exist, it will be created and filled with test data that comes from the above Test Library Inventory Table.

Encryption Checker File

This file “checker.txt” is created after first using the LMS software, and re-created if it does not exist.

When the LMS software is started, this file’s existence is checked, and, if not present, decryption of a non-existent or non-encrypted file will not carry out. If present, the encrypted USERFILE will be decrypted upon reading as normal.

The file helps determine whether the USERFILE is to be decrypted or not.

README File

Contains pertinent information for using the LMS software

Introduction

The Library Management System (LMS) provides an organized, interactive, and user-friendly way of accessing library content. An individual can access the LMS via an administrator username and password or a guest username and password. Users will be able to search through the library inventory via key search identifiers, including name of item(s), author of item(s), ISBN, etc. Additionally, guest users will be able to request for borrowing of item(s) for a specific amount of time, and view the history of borrowed item(s).

Project Description

The LMS begins with welcoming the individual to the *MEDKS Library*. The individual will then be prompted to select either administration login or guest login. If an individual is a new user and has no pre-existing account, the administrator can create an account for them. It is important to note that the administrator has the authority to add and remove any user account, but the guest does not have the same privilege. It is also important to note that if the administrator or guest were to enter the wrong username and/or password, a warning message will appear and notify the individual of an incorrectly entered username and/or password. All user credentials will be logged in a separate, accumulative text file that is encrypted when unused and decrypted when used.

Once properly logged into the library, the individual will be presented with an organized list of commands. Note, these commands vary depending on whether or not the individual is an administrator or guest to the library. If an administrator, the individual will be presented with the commands: CHANGE ADMIN USER/PASS, ADD GUEST USER, REMOVE GUEST USER, VIEW REQUESTED ITEMS, ACCEPT/REJECT BORROWING REQUESTS, ADD ITEM TO LIBRARY, EDIT ITEM IN LIBRARY, LIST USER ACCOUNTS, and EXIT. If a guest, the individual will be presented with the commands: SEARCH FOR ITEM, REQUEST TO BORROW ITEM, LIST LIBRARY ITEMS, VIEW BORROWING REQUESTS, and EXIT. It is important to note for borrow requests, the administrator user is permitted to accept or reject any borrow request.

To make the system more user-friendly, there are designated warning messages that will appear if the administrator tries to add a new item to the library with pre-existing information, if a user's search request returns null items, and if the guest tries to borrow more than three items. Additionally, to ensure information is stored and tracked properly, all borrowed history including borrow requests and library inventory will be logged in text files.

Users will be able to search through the library inventory via key search identifiers, such as name of item(s), author of item(s), ISBN, Format, etc. Additionally, Users will be able to request for borrowing of item(s) for an amount of time, and view the history of borrowed item(s).

As a security measure, all user information is encrypted. Temporary decryption will occur while the USERFILE is being read, as needed. Encryption is done by shifting the ASCII values of the characters in the .txt files by 100.

List of Key Global Variables

LIBRARYFILE
Defines the name of the .txt file containing the Library's contents (Default = "lmslibrary.txt")
USERFILE
Defines the name of the .txt file containing the User database contents (Default = "lmsusers.txt")
MAXITEMS
Defines the maximum amount of items in the Library (Default = 1000)
MAXSHELFLOCATION
Defines the maximum integer for the number of shelf locations in the Library (Default = 10000)
MAXREQUESTS
Defines the maximum amount of total requests from all users in the LMS (Default = 1000)
MAXUSERS
Defines the maximum amount of users in the LMS (Default = 100)
MAXITEMLENGTH
Defines the maximum length for strings referring to item qualities (Default = 41)
MAXITEMQTY
Defines the maximum quantity of an item in the Library (Default = 1000)
MAXITEMPRICE
Defines the maximum price of an item in the Library (Default = 100000)
MAXIDLENGTH
Defines the maximum length of usernames and passwords (Default = 15)
MINIDLENGTH
Defines the minimum length of passwords (Default = 8)
MAXBOOKSREQUESTED
Defines the maximum amount of requests made by a user (Default = 3)
MAXBOOKSLOANED
Defines the maximum amount of books to be loaned by a user (Default = 3)
ADMIN
Defines the string for what is placed in the .txt files when referring to Admins (Default = "Admin")
GUEST
Defines the string for what is placed in the .txt files when referring to Guests (Default = "Guest")

Key Local Variables

MaxAttempts
Placed in functions relating to user/admin login, describes the maximum amount of login attempts that could be made before the LMS software auto-closes (Default = 5)
FormatList[9][MAXITEMLENGTH]
A multidimensional array that stores the 9 possible item formats, "hardcover", "paperback", "journal", "dissertation", "magazine", "newspaper", "audiobook", "ebook", and "CD"

List of User Functions

Administrator Functions

changeAdminUser()
This function allows for the administrator to change administrator user's information.
addGuestUser()
This function allows for the administrator to add guest user information into the LMS.
deleteGuestUser()
This function allows for the administrator to delete guest user information from the LMS.
listRequests()
This function allows for the administrator to view an organized list of borrow requests.
processRequests()
This function allows for the administrator to process (accept / reject) borrow requests.
addItemtoLibrary()
This function allows for the administrator to add an item to the library.
editLibraryItem()
This function allows for the administrator to edit a pre-existing item in the library.
deleteItemfromLibrary()
This function allows for the administrator to delete a pre-existing item from the library.
listLibrary()
This function allows for the administrator to view an organized list of the library's inventory.
listUsers()
This function allows for the administrator to view a list of all user information.

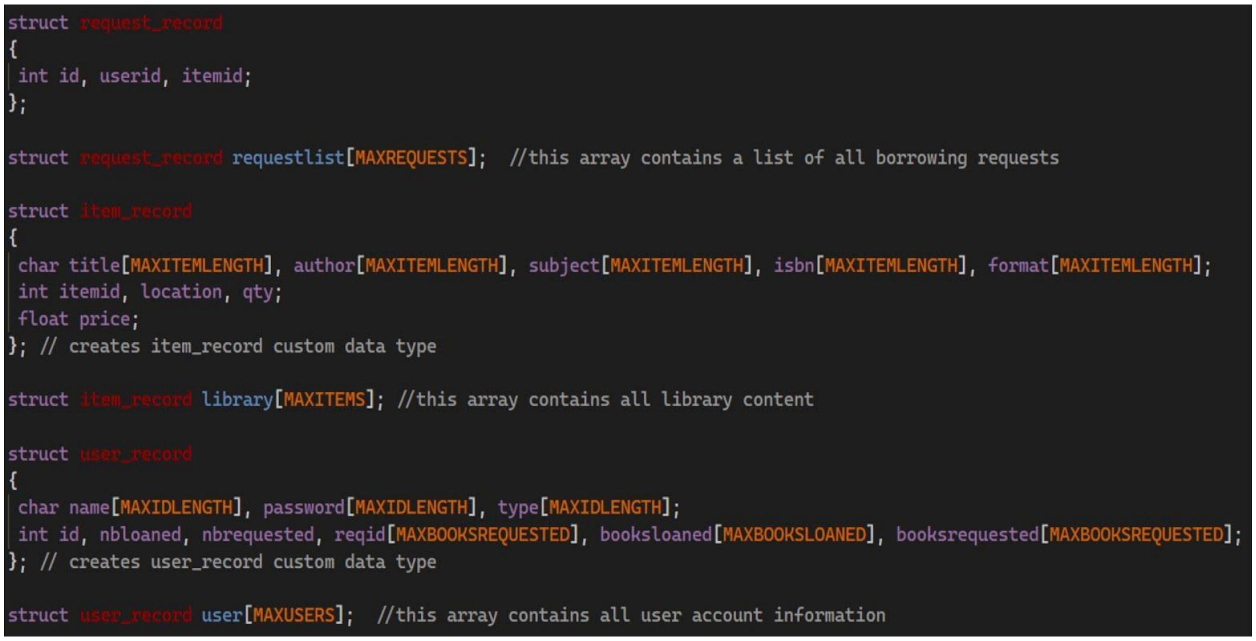
Guest Functions

searchForLibraryItem()
This function allows the guest to search for a library item via key search identifiers.
requestToBorrowLibraryItem()
This function allows the guest to request to borrow an item from the library.

Linked Functions

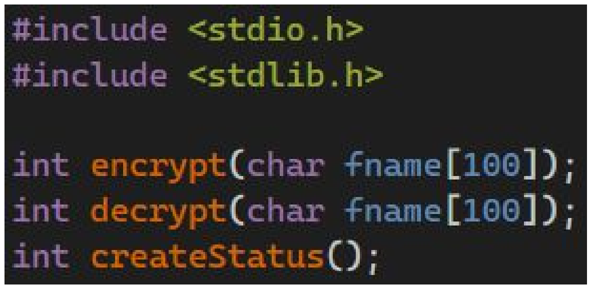
encrypt(char fname[100])
This function allows for the encryption of the contents of a file by shifting the ASCII values of all characters in the file forwards by 100 ("C Exercises: Encrypt").
decrypt(char fname[100])
This function allows for the decryption of the contents of a file by shifting the ASCII values of all characters in the file backwards by 100 ("C Exercises: Decrypt").
createStatus()
Creates and checks the existence of the Encryption Checker File described on page 5 ("Access(2)").

Featured Syntax



```c
struct request_record
{
 int id, userid, itemid;
};

struct request_record requestlist[MAXREQUESTS];  //this array contains a list of all borrowing requests

struct item_record
{
 char title[MAXITEMLENGTH], author[MAXITEMLENGTH], subject[MAXITEMLENGTH], isbn[MAXITEMLENGTH], format[MAXITEMLENGTH];
 int itemid, location, qty;
 float price;
}; // creates item_record custom data type

struct item_record library[MAXITEMS]; //this array contains all library content

struct user_record
{
 char name[MAXIDLENGTH], password[MAXIDLENGTH], type[MAXIDLENGTH];
 int id, nbloaned, nbrequested, reqid[MAXBOOKSREQUESTED], booksloaned[MAXBOOKSLOANED], booksrequested[MAXBOOKSREQUESTED];
}; // creates user_record custom data type

struct user_record user[MAXUSERS];  //this array contains all user account information
```

The use of structures to create custom data types for storing user and library item information



```c
#include <stdio.h>
#include <stdlib.h>

int encrypt(char fname[100]);
int decrypt(char fname[100]);
int createStatus();
```

The contents of the encrypt.h header file, linking encrypt.c with the encryption/decryption functions and LMS.c with admin/user functions, initialized with #include “encrypt.h” in LMS.c.

General Findings

While previous lectures introduced a brief concept of encryption and decryption via Caesar’s cipher, a new concept the group learned was how to encrypt and decrypt a whole text file rather than individual characters in a single line of code. Researching and learning about this new method of encryption and decryption allowed us to properly protect the information stored in text files in an orderly fashion.

The research we carried out to form this project not only improved the security for the User’s data, but to make our coding challenge slightly easier. In programming, researching new documentation and examples on how to complete certain tasks is a must to learn how to properly implement them yourself, and our group certainly gained knowledge from the research we had to do to translate the ideas in our mind to C code.

References

"Access(2) - Linux Man Page." *Die.net*, linux.die.net/man/2/access.

"C Exercises: Encrypt a Text File." *Atlantic.net*, w3resource, 26 Feb. 2020, www.w3resource.com/c-programming-exercises/file-handling/c-file-handling-exercise-13.php.

"C Exercises: Decrypt a Previously Encrypted File." *Atlantic.net*, w3resource, 26 Feb. 2020, www.w3resource.com/c-programming-exercises/file-handling/c-file-handling-exercise-14.php.