



Programmierung I

Beleg

Falk-Jonatan Strube

Inhaltsverzeichnis

1	Persönliche Angaben	3
2	Aufgabenstellung	3
3	Anforderungsanalyse	3
4	Implementation	4
5	Kompilieren	4
6	Doc	5

1 Persönliche Angaben

Name: Falk-Jonatan Strube

Matrikelnr.: 39467

Aufgabenstellung: 4 (Verleih-Bibliothek) $(39467 \% 4) + 1 = 4$

Bedienoberfläche: 3 (Web-CGI) $(39467 \% 3) + 1 = 3$

2 Aufgabenstellung

Programmieren Sie ein Programm zur Verwaltung verborgter Bücher, CDs, DVDs (Medien). Dabei sollen Name, Autor/Interpret (kann auch leer bleiben) und leihende Person (Name) verwaltet werden. Wird ein Medium zurückgegeben, ist der Datensatz zu entfernen.

Es soll sowohl nach dem Medium, als auch nach der leihenden Person gesucht werden können.

3 Anforderungsanalyse

1.) C-Programm bestehend aus mindestens 3 Modulen

- Modul bestehend aus .c und .h Datei
- Module müssen getrennt compilierbar sein

2.) Daten in geeigneter Datenstruktur speichern

- im Quelltext geeignete Struktur
- extern in geeigneter Struktur nach Beendigung des Programms abspeichern

3.) Daten bearbeiten

- Daten erfassen
- Daten löschen
- Daten suchen
- Tabellarische Anzeige aller Daten
- Einzelansicht gesuchter Daten

4.) Quelltext-Dokumentation

- Funktionen
- Urheberschaft im Quelltextkopf

5.) Speicherbereiche in erforderlicher Größe bereitstellen und wieder freigeben

6.) Programm muss auf Rechnern im Rechnerlabor der HTW übersetzt und vorgeführt werden.

4 Implementation

1.) Module:

Modul	Nutzen	Beschreibung
lendlib	Hauptmodul	
lendlibin	Input	Eingabeverarbeitung aus Webform/Terminal und Datei
lendlibout	Output	Ausgabe über Webform/Terminal und Datei
lendlibitem	Medium	Verwaltung Ausleihmedium

2.) Datenstruktur

- Struct für Medien
 - ID
 - Medienart (enum: Buch, CD, DVD, andere) [optional]
 - Titel des Mediums (Name)
 - Autor/Interpret des Mediums [optional]
 - Name der Person, dem das Medium verliehen wurde
- Abspeichern in .csv

3.) Bearbeitung:

- Daten erfassen: einfügen in doppelt verkettete Ringliste
- Daten löschen: löschen aus doppelt verketteter Ringliste
- Daten sortieren (nach Mediumtitel und Verleihname)
- Daten suchen (nach Mediumtitel und Verleihname)
- Anzeige der Daten: siehe Quelltext

4.) Quelltext-Dokumentation:

siehe Quelltext

5.) Speicherverwaltung:

siehe Implementation der Datenverarbeitung

6.) Lauffähigkeit Rechnerlabor:

siehe Vorführung

5 Kompilieren

Kommando	Ergebnis
gcc -o cgi-bin/lendlib.cgi -D CGI *.c	für Web-CGI kompilieren
gcc -o lendlib -D DEBUG *.c	Anzeigen von Debug-Meldungen
gcc -o lendlib -D DCOLOR *.c	Farbige Anzeige im Terminal
gcc -o lendlib -D DPRE *.c S	Möglichkeit des Verlangsamten, anschaulichen Ablaufes

Achtung: Das CGI-Programm muss aufgrund Dateiabhängigkeiten in Ordner „cgi-bin“ liegen (sodass bpsw. Ordner „cgi“ oder die „.csv“-Datei in „..“ liegen)!

Empfohlene Anweisungen:

- gcc -DDEBUG -DDCOLOR -o lendlib *.c && ./lendlib
- gcc -DCGI -o cgi-bin/lendlib.cgi *.c && sensible-browser localhost/cgi-bin/lendlib.cgi

6 Doc

Kapitel 1

Modul-Verzeichnis

1.1 Module

Hier folgt die Aufzählung aller Module:

LendLib	5
LendLibIn	6
LendLibItem	9
LendLibOut	15

Kapitel 2

Klassen-Verzeichnis

2.1 Auflistung der Klassen

Hier folgt die Aufzählung aller Klassen, Strukturen, Varianten und Schnittstellen mit einer Kurzbeschreibung:

LITEM	Eintrag in der Liste der verliehenen Medien	19
medium	Das ausgeliehene Medium	19
theLib	Verwaltet die Liste der verliehenen Medien	20

Kapitel 3

Modul-Dokumentation

3.1 LendLib

Hauptmodul der Ausleihbibliothek.

Funktionen

- int `main` (int argc, char *argv[])
Die Main-Funktion.

3.1.1 Ausführliche Beschreibung

Hauptmodul der Ausleihbibliothek.

3.1.2 Dokumentation der Funktionen

3.1.2.1 int main (int argc, char * argv[])

Die Main-Funktion.

- Ausgabevariablen vorbereiten
- Kopfzeilen ausgeben
- Einträge aus Datei einlesen (falls sie existiert)

CGI-Ausgabe:

Terminalausgabe:

- aus Datei erhaltene Medien anzeigen
- Nutzerinput verarbeiten

beide:

- Daten in Datei speichern
- Speicher freigeben
- Fußzeilen ausgeben

3.2 LendLibln

Funktionen für jegliche Art von Ausgabe.

Makrodefinitionen

- `#define MALLOCERR "Fehler bei der Speicherzuweisung"`

Funktionen

- void `readfile` ()
Liest Verleihmediendaten aus Datei aus.
- int `getSize` (FILE *libdb)
Gibt Größe der gespeicherten CSV-Datenbank aus.
- void `getInput` ()
Verarbeitet Nutzerinput im Terminal.
- void `getPost` ()
Verarbeitet POSTs des CGI.
- char * `strtok2` (char *str, char const *delims)
Eigenes strtok, welches auch leere Einträge ausgibt.

Variablen

- FILE * **libdb**
- char **vbuf** [128]
- const char * **filename**
- char **posted** [256]
- char **pTemp1** [256]
- char **pTemp2** [256]
- char **pTemp3** [256]

3.2.1 Ausführliche Beschreibung

Funktionen für jegliche Art von Ausgabe.

3.2.2 Dokumentation der Funktionen

3.2.2.1 void getInput ()

Verarbeitet Nutzerinput im Terminal.

Hilfezeilen mit Anweisungen anzeigen

Dateien noch einmal auflisten

Ausleihmedium hinzufügen

- Eingabe Medienart
- (Festlegung der Medienart durch entsprechende Zahl aus mType oder Anfangsbuchstabe: Buch, Cd, Dvd)
- (Titel, Interpret/Autor und Ausleihender eingeben lassen) und jeweils in freigegebenen Speicher schreiben
- Eingebene Daten als Medium speicher und in Liste einfügen

Medien sortieren, dazu nach Sortierkategorie fragen

Medium löschen, dazu ID abfragen

Medien finden, dazu nach Suchkategorie (Titel oder Ausleihender) und der zu suchenden Zeichenkette fragen

Ansonsten Programm beenden

3.2.2.2 void getPost ()

Verarbeitet POSTs des CGI.

Der POST wird jeweils mit [strtok2](#) auseinandergeschnitten und verarbeitet. Die vorgehensweise ist ähnlich wie bei der Terminaleingabe. POST einlesen

POSTs unterscheiden sich durch den Anfangsbuchstaben (wie bei Terminaleingabe)

- Sucheingabe verarbeiten
- Etwas finden
- Medium einfügen
- (Funktioniert wie bei Terminaleingabe: Es wird jeweils Speicher bereitgestellt und Angaben darin geschrieben.
- Löschen

3.2.2.3 int getSize (FILE * libdb)

Gibt Größe der gespeicherten CSV-Datenbank aus.

Parameter

<i>libdb</i>	Eine Datei
--------------	------------

Rückgabe

Anzahl von Linien (also Einträgen)

Quelle: <http://stackoverflow.com/a/1910795>

- Anzahl der Zeilenumbrüche zählen
- Im Fall, dass die letzte Zeile keinen Umbruch enthält oder leer ist (bis auf den Umbruch), wird diese dazu gezählt oder eben nicht

3.2.2.4 void readfile ()

Liest Verleihmediendaten aus Datei aus.

Je nach Terminal- oder CGI-Anwendung ist der Dateipfad anders

- Datei öffnen
- Datei Zeilenweise durchgehen und als Medium in Liste einfügen

3.2.2.5 char * strtok2 (char * *str*, char const * *delims*)

Eigenes strtok, welches auch leere Einträge ausgibt.

Parameter

<i>str</i>	Der String
<i>delims</i>	Die Trennzeichen Quelle: http://stackoverflow.com/a/8706031

3.3 LendLibItem

Funktionen und Deklarationen rund um ausgeliehene Medien.

Klassen

- struct `medium`
Das ausgeliehene Medium.
- struct `LITEM`
Eintrag in der Liste der verliehenen Medien.
- struct `theLib`
Verwaltet die Liste der verliehenen Medien.

Typdefinitionen

- typedef struct `LITEM` `Item`
Eintrag in der Liste der verliehenen Medien.

Aufzählungen

- enum `mType` { `other`, `book`, `cd`, `dvd` }
Typ des ausgeliehenen Mediums.
- enum `sBy` { `title`, `lender`, `id` }
Möglichkeiten, nach denen Sortiert/gesucht werden kann.

Funktionen

- `medium * createltem` (int ntype, char *ntitle, char *nauthor, char *nlender)
Erstellt einen neuen Medieneintrag.
- `medium * createltemF` (FILE *libitem)
Erstellt neuen Medieneintrag aus Dateizeile.
- void `insertItem` (`medium *nMedium`, `theLib *inLib`)
Fügt Medium in Liste ein.
- char * `getmType` (int type)
Gibt String für enum `mType` aus.
- void `deleteltem` (int theID, `theLib *inLib`)
Löscht Medium aus der Liste.
- void `sortItems` (`sBy` sortBy, `theLib *inLib`)
Sortiert die Liste nach Mediumtitel/Leihendem.
- void `findItem` (char *sItem, `sBy` findBy, `theLib *inLib`)
Suche, bzw. finde ein ausgeliehenes Medium.
- void `freeAll` (`theLib *inLib`)
Den gesamten Speicher freigeben.
- int `stricmp` (char const *a, char const *b)
Vergleicht zwei "Strings" unabhängig von Groß- oder Kleinbuchstaben Quelle: <http://stackoverflow.com/a/5820991>.
- void `initLib` ()

Variablen

- `theLib` * `currLib`
- `theLib` `myLib`
- `lItem` * `myItems`
- `medium` * `myMedia`
- `int` `myItemsCount`
- `int` `myMediaCount`
- `unsigned int` `idc`

3.3.1 Ausführliche Beschreibung

Funktionen und Deklarationen rund um ausgeliehene Medien.

3.3.2 Dokumentation der Aufzählungstypen

3.3.2.1 `enum mType`

Typ des ausgeliehenen Mediums.

Aufzählungswerte

- other*** Medium ohne Kategorie.
- book*** Bücher.
- cd*** CDs.
- dvd*** DVDs.

3.3.2.2 `enum sBy`

Möglichkeiten, nach denen Sortiert/gesucht werden kann.

Aufzählungswerte

- title*** Nach Medientitel.
- lendeer*** Nach Leihendem.
- id*** Nach ID.

3.3.3 Dokumentation der Funktionen

3.3.3.1 `medium * createltem (int ntype, char * ntitle, char * nauthor, char * nlendeer)`

Erstellt einen neuen Medieneintrag.

Parameter

<i>ntype</i>	Typ des Mediums
<i>ntitle</i>	Titel des Mediums
<i>nauthor</i>	Autor/Interpret des Mediums
<i>nlendeer</i>	Person, dem das Medium ausgeliehen wurde

Rückgabe

Gibt das erstellte **medium** zurück

- ID festlegen, dazu wird die Liste nach IDs sortiert und nach der ersten "Zahlenlücke" gesucht
- Type festlegen
- Titel festlegen
- Interpret/Autor festlegen
- Leihendem festlegen

3.3.3.2 **medium * createItemF (FILE * libitem)**

Erstellt neuen Medieneintrag aus Dateizeile.

Parameter

<i>libitem</i>	Die Datei
----------------	-----------

Rückgabe

Gibt das erstellte **medium** zurück

- ID festlegen
- Typ festlegen
- Titel festlegen
- Interpret/Autor festlegen
- Leihendem festlegen

3.3.3.3 **void deleteltem (int theID, theLib * inLib)**

Löscht Medium aus der Liste.

Parameter

<i>theID</i>	Das zu löschende Medium (ID)
<i>inLib</i>	in dieser Liste

- Medium mit ID finden
- Medium aus Liste löschen indem Zeiger der Nachbarn neu gesetzt werden und anschließend der Speicher freigegeben wird.

3.3.3.4 void findItem (char * sItem, sBy findBy, theLib * inLib)

Suche, bzw. finde ein ausgeliehenes Medium.

Parameter

<i>sItem</i>	Nach dieser Zeichenkette soll gesucht werden
<i>findBy</i>	Die Zeichenkette soll in diesem Eintrag des ausgeliehen Mediums gesucht werden (sBy)
<i>inLib</i>	in dieser Liste

Rückgabe

Gibt das gefundene Medium zurück

- Durch myLib [theLib](#) gehen und alle gefunden in eine neue, temporäre [theLib](#) Liste einfügen, die dann ausgegeben wird.

3.3.3.5 void freeAll (theLib * inLib)

Den gesamten Speicher freigeben.

Parameter

<i>inLib</i>	in dieser Liste
--------------	-----------------

3.3.3.6 char * getmType (int type)

Gibt String für enum [mType](#) aus.

Parameter

<i>type</i>	Zahl von mType
-------------	--------------------------------

Rückgabe

[mType](#) als String

3.3.3.7 void insertItem (medium * nMedium, theLib * inLib)

Fügt Medium in Liste ein.

Parameter

<i>iMedium</i>	Das einzufügende Medium
----------------	-------------------------

- Listenobjekt erstellen und Inhalt zuweisen

Je nach Sortierung der Liste, muss anders eingefügt werden, falls nach Titel sortiert:

- Falls Liste noch leer, diese initialisieren
- Falls Liste einelementig ist, das nächste Element einfach einfügen
- Alle anderen Elemente sortiert in Liste einfügen
- - Nach einem Element suchen, welches größer ist, damit man das einzufügende Element vor diesem einfügen kann.
- - Wenn größeres vorm Listenende gefunden wurde, Element davor einfügen
- - Ansonsten Element am Listenende je nach größe davor oder danach einfügen.

falls nach Ausleihendem sortiert:

- Falls Liste noch leer, diese initialisieren
- Falls Liste einelementig ist, das nächste Element einfach einfügen
- Alle anderen Elemente sortiert in Liste einfügen
- - Nach einem Element suchen, welches größer ist, damit man das einzufügende Element vor diesem einfügen kann.
- - Wenn größeres vorm Listenende gefunden wurde, Element davor einfügen
- - Ansonsten Element am Listenende je nach größe davor oder danach einfügen.

Falls nach ID sortiert:

- Falls Liste noch leer, diese initialisieren
- Falls Liste einelementig ist, das nächste Element einfach einfügen
- Alle anderen Elemente sortiert in Liste einfügen
- - Nach einem Element suchen, welches größer ist, damit man das einzufügende Element vor diesem einfügen kann.
- - Wenn größeres vorm Listenende gefunden wurde, Element davor einfügen
- - Ansonsten Element am Listenende je nach größe davor oder danach einfügen.

3.3.3.8 void sortItems (sBy sortBy, theLib * inLib)

Sortiert die Liste nach Mediumtitel/Leihendem.

Parameter

sortBy	Nach was sortiert werden soll (sBy)
inLib	in dieser Liste

Zum Sortieren wird eine [medium](#) Hilfsliste erstellt, in die alle Medien aus der [theLib](#) Originalliste kopiert werden, um sie dann wieder in die Originalliste einzufügen(dabei sortiert ::insertItem schon).

3.4 LendLibOut

Funktionen für jegliche Art von Ausgabe.

Aufzählungen

- enum `pType` {
 `in`, `out`, `error`, `status`,
 `extra` }
 Verschieden Ausgabemöglichkeiten.

Funktionen

- void `prepareOut` ()
 Bereitet Variablen für den Output vor.
- void `printItems` ()
 Gibt die Liste der ausgeliehenen Medien aus.
- void `libprint` (`pType` type, const char *printable,...)
 Terminalausgabe abhängig von (beim kompilieren) definierten Paramtern.
- void `setColor` (int pType)
 Setzt die Terminal-Schriftfarbe abhängig vom Typ.
- void `resetColor` ()
 Setzt die Terminal-Schriftfarbe zurück auf den Standard.
- void `printHead` ()
 Gibt einen Header aus.
- void `printHTMLInter` ()
 Gibt die HTML-Bedienelemente aus.
- void `printFoot` ()
 Gibt einen Footer aus.
- void `printTLine` (char type, int length)
 Gibt eine Linie im Terminal aus.
- void `saveDBtoFile` ()
 Speichert Liste in Datei.
- void `sleep_ms` (int milliseconds)
 Cross-Platform Sleep Funktion in Millisekunden.

3.4.1 Ausführliche Beschreibung

Funktionen für jegliche Art von Ausgabe.

Ausgabe abhängig von den übers kompilieren definierten Parameter:

Parameter

<code>CGI</code>	für eine .cgi-Datei
<code>DCOLOR</code>	für eingefärbte Terminalausgaben

3.4.2 Dokumentation der Aufzählungstypen

3.4.2.1 enum ptype

Verschieden Ausgabemöglichkeiten.

Aufzählungswerte

- in** folgende Eingabe
- out** normale Ausgabe
- error** Fehlermeldungen.
- status** Statusmeldungen.
- extra** Fehlerbehebung.

3.4.3 Dokumentation der Funktionen

3.4.3.1 void libprint (ptype type, const char * printable, ...)

Terminalausgabe abhängig von (beim kompilieren) definierten Paramtern.

Parameter

type	Ausgabety (ptype)
------	-------------------

Quelle: <http://www.ozzu.com/cpp-tutorials/tutorial-writing-custom-printf-wrapper-function-t.html>

- Im Falle CGI werden die Ausgaben in verschieden Formatierte divs gefüllt
- Im Terminal wird je nach definierten Strings etwas mit oder Farbe ausgegeben - oder nicht

Falls Fehlerausgabe deaktiviert ist, keine Fehler ausgeben!

3.4.3.2 void prepareOut ()

Bereitet Variablen für den Output vor.

- Berechnung der Terminalbreite (Quelle: <http://stackoverflow.com/a/1022961>)

3.4.3.3 void printFoot ()

Gibt einen Footer aus.

Im Terminal ein Abschlusstrich, im CGI den HTML-Footer

3.4.3.4 void printHead ()

Gibt einen Header aus.

Im Terminal ein paar erste Zeilen, im CGI den HTML-Header

3.4.3.5 void printItems ()

Gibt die Liste der ausgeliehenen Medien aus.

Im CGI durch eine Tabelle (in der auch schon Bedienelemente fürs löschen, sortieren usw. eingeschlossen sind)

Im Terminal in einer Tabellenimitation

3.4.3.6 void printTLine (char *type*, int *length*)

Gibt eine Linie im Terminal aus.

Parameter

<i>type</i>	Ein char, der die Linie visuell bestimmt
<i>length</i>	Die Länge der Linie, bei der Länge 0 wird die Breite des Terminals

3.4.3.7 void resetColor ()

Setzt die Terminal-Schriftfarbe zurück auf den Standard.

Nur, falls beim Compilieren DCOLOR definiert wurde.

3.4.3.8 void setColor (int *pType*)

Setzt die Terminal-Schriftfarbe abhängig vom Typ.

Parameter

<i>pType</i>	Ausgabetyp (ptype)
--------------	--------------------------------------

Nur, falls beim Compilieren DCOLOR definiert wurde.

3.4.3.9 void sleep_ms (int *milliseconds*)

Cross-Platform Sleep Funktion in Millisekunden.

Quelle: <http://stackoverflow.com/a/28827188>

Kapitel 4

Klassen-Dokumentation

4.1 LITEM Strukturreferenz

Eintrag in der Liste der verliehenen Medien.

```
#include <lendlibitem.h>
```

Öffentliche Attribute

- struct `LITEM` * `next`
Zeigt auf vorhergehenden Eintrag.
- struct `LITEM` * `prev`
Zeigt auf folgenden Eintrag.
- `medium` * `item`
Inhalt.

4.1.1 Ausführliche Beschreibung

Eintrag in der Liste der verliehenen Medien.

Die Dokumentation für diese Struktur wurde erzeugt aufgrund der Datei:

- `htw/Programmierung 1/Beleg/lendlibitem.h`

4.2 medium Strukturreferenz

Das ausgeliehene Medium.

```
#include <lendlibitem.h>
```


Öffentliche Attribute

- int **id**
- mType type
Typ des ausgeliehen Mediums.
- char * **title**
Titel des Mediums.
- char * **author**
Autor bzw. Interpret des Mediums.
- char * **lendeer**
Name der Person, dem das Medium verliehen wurde.

4.2.1 Ausführliche Beschreibung

Das ausgeliehene Medium.

Die Dokumentation für diese Struktur wurde erzeugt aufgrund der Datei:

- htw/Programmierung 1/Beleg/lendlibitem.h

4.3 theLib Strukturreferenz

Verwaltet die Liste der verliehenen Medien.

```
#include <lendlibitem.h>
```

Öffentliche Attribute

- lItem * **first**
Erster Eintrag.
- lItem * **curr**
aktueller Eintrag
- unsigned int **size**
größe der Liste
- unsigned int **sort**
Liste ist sortiert nach 0: Titel, 1:Ausleihenden.

4.3.1 Ausführliche Beschreibung

Verwaltet die Liste der verliehenen Medien.

Die Dokumentation für diese Struktur wurde erzeugt aufgrund der Datei:

- htw/Programmierung 1/Beleg/lendlibitem.h