ООП 2021/2022 - Проект 1

Тема 2 – Електронна библиотека

Документация

В същността си тази програма представля съвкупност от операции, които моделират една база данни в зависимост от потребителските параметри. Програмата трябва успешно да зарежда и да записва електронната библиотека съответно от и на твърдата памет, да осъществява подходящ потребителски интерфейс, както и успешно и ефективно да изпълнява зададените от потребителя команди. Не се изискват сложни алгоритми, единствено правилно управление на динамичната памет е необходимо.

Проектът се състои от 4 основни компонента – функциите в main.cpp, и класовете Library, Book, PasswordManager. Функциите в main.cpp осъществяват потребителския интерфейс, както и връзката с електронната библиотека, Library осъществява самата библиотека, както и командите, които могат да се прилагат върху нея. Book описва една книга от библиотеката, а PasswordManager управлява паролите за оторизиран достъп. Изпълнението на програмата минава през следните етапи:

1) Зареждане на библиотеката и паролите в паметта от файловете

2) Получаване на заявки от потребителя и изпълнението им.

3) Записване на библиотеката и паролите във файлове, освбождаване на динамичната памет и прекратяване на програмата.

Следва представяне на компонентите:

Файлът Book.h съдържа, освен класа Book, няколко помощни функции, няколко избриоми типа и константи. Константите определят максималната дължина на низовете, които характеризират една книга.

void write\_bintext(ofstream&f,const char\* низ); - тази функция записва низа в двоичния файл, свързан с потока f. Първо записва дължината на низа+1, а след това и самия низ, заедно с терминиращата нула.

enum SearchMode{title\_m,author\_m,isbn\_m,description\_m}; - тип изброим, който характеризира режим на търсене.

enum SortMode{title\_s,author\_s,rating\_s}; - тип изброим който характеризира режим на сортиране.

enum PrintMode{sent\_mode,row\_mode}; - тип изброим, характеризиращ режима на отпечатване на съдържанието на една книга на екрана.

bool str\_eq(const char\*,const char\*,bool ignore = false); - връща true ако двата стринга - параметъри са равни, false – иначе. Параметърът ignore определя дали при сравнение да се игнорира регистъра на буквите.

Класът Book се състои от следните член-данни с частен достъп:

char \*author,\*heading,\*directory,\*description,\*isbn;

Те представляват указател към динамична памет, където е записан низ със съответната информация за книгата.

int rating; рейтънгът на книгата

bool loaded; true – ако указателите по-горе сочат към динамична памет, false – ако са nullptr*.*

Класът притежава и една функция с вътрешен достъп

void deleteDyn(); - изтрива динамичната памет, сочена от указетилте по-горе.

Следващите член-функции имат публичен достъп:

Конструкторът по подразбиране на Book присвоява на указателите стойността nullptr, а на другите полета – стойността 0.

Останалите конструктори са стандартните копиращ конструктор и преместващ конструктор, а операторите за присвояване са също стандартни и се обръщат към функцията deleteDyn() за освобождаване на паметта на присвояващия обект. Деструкторът съдържа единствено обръщение към същата функция.

bool load(ifstream& str); - тази функция приема параметър поток, свързан с двоичен файл с информация за книгата. Зарежда тази информация в обекта като заделя динамична памет. Връща true ако информацията е успешно прочетена, false – иначе.

void setField(fieldType); - тези функции са стандартни мутатори за полето Field, като функцията присвоява параметъра си на полето.

bool compBook(const char\* низ,SearchMode,bool ignore); - Връща true ако низа-параметър съвпада с поле на обекта, определено от SearchMode, false – иначе. Параметърът ignore определя дали да се игнорира регистърът на буквите при сравнението.

bool lessThan(const Book& book,SortMode); - връща true ако низа, сочен от полето на обекта, оперделено от SortMode е лексикографски по-малък от низа, сочен от съответното поле на параметъра book.

void print(); - отепчатва на екрана пълна информация за книгата

void shortPrint(); - отпечатва на екрана полетата heading, author, isbn.

bool getFile(ifstream&); - Ако файлът с директория directory съществува, функцията връща true и свързва потока с файла. False – иначе.

bool delFile(); - Ако файлът с директория directory съществува, функцията връща true и изтрива файла. False – иначе.

void save(ofstream& file) – Записва низовете, сочени от полетата на обекта, в двоичния файл file, използвайки функцията write\_bintext.

Файлът Library.h съдържа класа Library

Класът Library има следните полета:

Book \*list; - указател към масив в динамичната памет от тип Book – това са книгите в електронната библиотека

int book\_amount; - броя на книгите в ел. Библиотеката или големината на масива list.

Всички член-функции на класа са с публичен достъп:

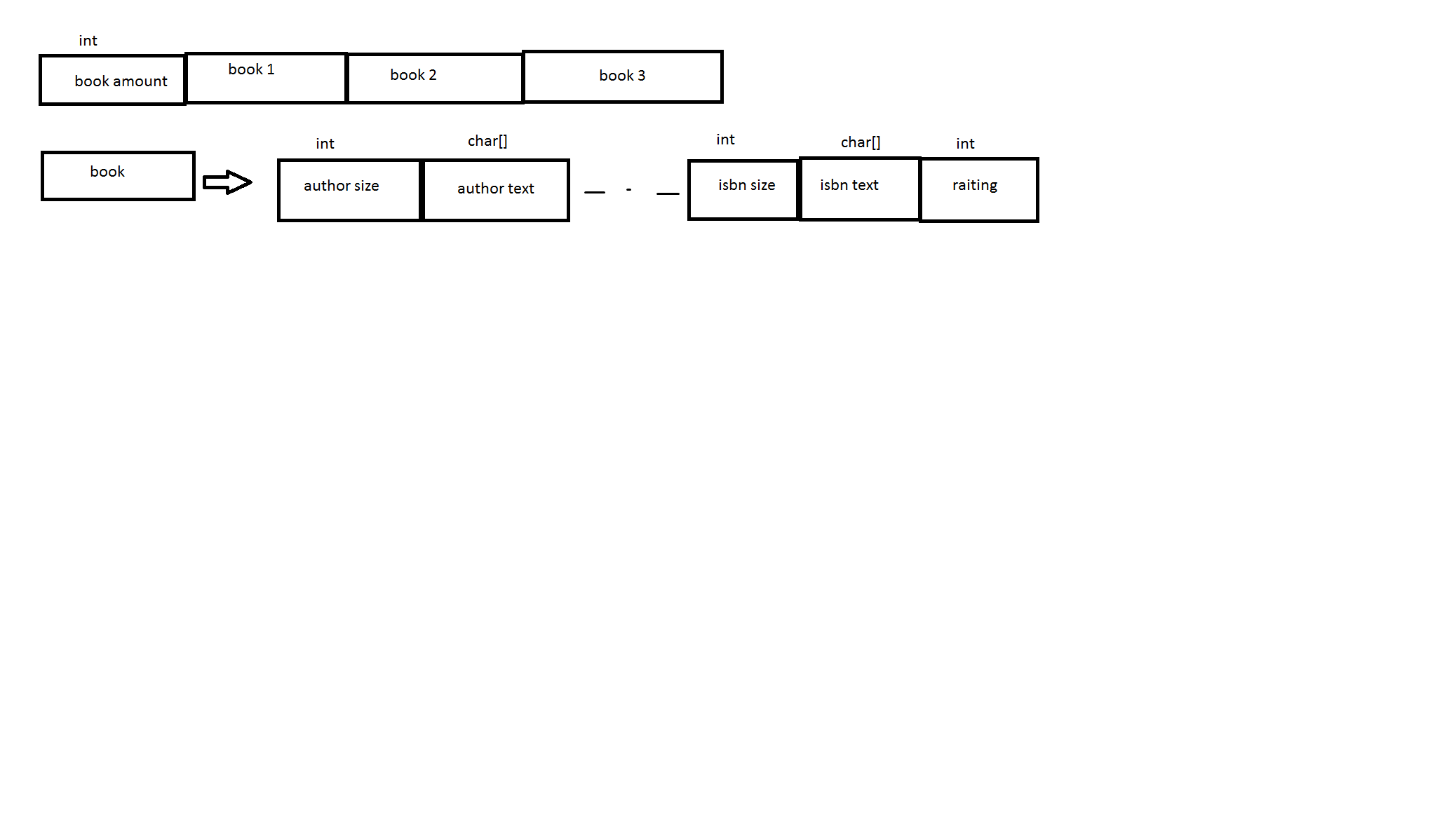
Library(); - конструктор по подразбиране – присвоява nullptr на list и 0 на book\_amount.

Library(const Library&) = delete;

Library& operator=(const Library&) = delete; - тези конструктор и оператор за присвояване са изтрити, защото програмата се нуждае само от един обект от тип Library.

~Library(); - обикновен деструктор, изтрива масива в динамичната памет list.

bool load(ifstream& file); - зарежда цялата електронна библиотека от двоичния файл file. За всяка книга се вика функцията Book::load(file). Информацията за книгите се съхранява във файла в следния формат(фиг.1):



Фиг.1

void add(const Book &); - Добавя книга към библиотеката. Създава нов масив и презаписва всичко от стария в новия, заедно с новата книга. Използва std::move.

bool findBook(Book& book,const char\* низ,SearchMode,bool ignore); - Търси книга в библиотеката, която има поле, равно на низа-параметър. Полето се определя от SearchMode. Параметърът ignore определя дали да се игнорира регистъра на буквите при сравнение. Намерената книга се присвоява на book и функцията връща true. Ако функцията не отркие книгата, връща false. Вътрешно се извиква Book::compBook за всяка книга

void simplePrint(); - изкарва на екрана пълна информация за всички книги в библиотеката, без да са сортирани

void printSorted(SortMode mode,bool des); - изкарва на екрана книгите, подредени лексикографски по поле, определено от SearchMode. Параметърът des определя дали да са в низходящ(true) или възходящ(false) ред

void delBook(const char \* heading); - изтрива книгата със заглавие heading от библиотеката. Презаписва целия масив list, като използва std::move. Ако такава няма, функцията не прави нищо.

void save(ofstream& file); - записва информацията за цялата библиотека в двоичния файл file.

Файлът PasswordManager.h съдържа една константа max\_pass – максималната дължина на всяка от паролите.

Класът PasswordManager съдържа следните полета:

char \*\*arr; - указател към масив от указатели в динамичната памет. Всеки елемент на този масив сочи към низ в динамичната памет, който представлява една от паролите.

int size; - броя на паролите в arr.

Класът съдържа следните публични член-функции:

PasswordManager(); - конструктор по подразбиране, инициализира двете полета с nullptr и 0 съответно

PasswordManager(const PasswordManager&) = delete;

PasswordManager& operator=(const PasswordManager&) = delete; - не трябва да има други обекти от този клас.

~PasswordManager(); - деструкторът освобождава динамичната памет за паролите, както и масивът arr.

bool load(ifstream& file); - зарежда паролите в arr от файла file. Файлът е текстов, първо е записан броя на паролите, а след това всяка парола на отделен ред. При успешно зареждане връща true, иначе false.

bool find(const char \* низ); - връща true ако параметърът-низ съвпада с някоя от паролите в arr. Иначе false.

void addPass(const char \*); - добавя нова парола в arr.

bool save(ofstream& file); - записва паролите в текстовия файл file.

Потребителският интерфейс се осъществява във файла main.cpp. Той съдържа следните константи:

const char \* const lib\_dir - директорията и името на файла с електронната библиотека

const char \* const pass\_dir – директорията и името на файла с паролите.

Функцията main() единствено вика run().

void loadLib(Library &lib); - зарежда електронната библиотека от файл. Ако файлът не съществува, го създава. Извиква Library::load

void loadMgr(PasswordManager &mgr); - зарежда паролите от файл. Ако файла не съществува, функцията създава файл с една парола – admin и зарежда него. Извиква mgr.load.

void run(); - основната функция изпълняваща програмта. В нея има безкраен цикъл, в който се вика mainPage(). Той се прекратява единствено при такава команда на потребетиля.

int mainPage(bool &admin, Library &lib, PasswordManager &mgr) – обработва една заявка на потребителя, като евентуално променя lib и/или mgr. Параметърът admin, определя дали потребителят има администраторски права. Връща номера на заявката.

void clearCin(); - помощна функция. Изтрива флаговете на cin и премахва всички символи от потока до срещане на нов ред включително.

bool checkCin(); - връща true ако cin няма флагове за грешка. Иначе вика clearCin(), отпечатва съобщение за грешка и връща false.

bool checkCommand(const char \* command,char minval,char maxval) – Проверява дали command се състои от един символ със стойност между minval и maxval.

void saveLib(Library& lib) – запазва информацията от библиотеката във файл. Извиква Library::save.

void savePass(PasswordManager &mgr) – записва паролите във файл. Извиква mgr.save.

void printSorted(Library &lib) – Приема заявка от потребителя за вида сортиране. След това отпечатва съдържанието на библиотеката сортирано. Извиква lib.printSorted

void findBook(Library& lib) – Приема текст от потребителя и търси такава книга в библиотеката съгласно всички възможни критерии. Извиква lib.findBook

void wait() – Приема нов ред от потребителя без да прави нищо с него.

void printContent(Library& lib) – Отпечатва съдържанието на книгата съгласно посочен от потребителя режим. Очаква от потребителя да въведе заглавието на търсената книга. Извиква Lib.findBook и Book::getFile.

void login(bool &admin, PasswordManager &mgr) – приема парола от неоторизиран потребител. Ако такава парола съществова в mgr, дава оторизиран достъп на потребителя. Извиква mgr.find.

void addBook(Library &lib) – Добавя книга в библиотеката, като приема всички данни за нея от потребителя. Извиква lib.add.

void removeBook(Library &lib) – Премахва книга от библиотеката, според въведено от потребителя заглавие. Извиква lib.findBook, lib.delBook и Book::delFile.

void addUser(PasswordManager &mgr) – Добавя нова парола, въведена от потребителя, в mgr. Извиква mgr.addPass.

Идеи за подобрения:

1) За подобряване на ООП архитектурата е възможно потребителския интерфейс да се изведе в отделен клас с полета lib и mgr.

2) Възможно е масивът с книгите да бъде динамичен, което ще оптимизра операциите за добавяне и махане на книга. Друг вариант е да се използва класа vector.

3) Да се направи графичен интерфейс, или съдържанието на конзолата да се изтрива при всяка нова заявка на потребителя.

4) Да се направи някаква защита на достъпа до файловата система за да не може програмата да трие или обработва чужди файлове.

5) Програмата да копира текстовите файлове със съдържанията на книгите в отделна папка

6) Да се даде възможност на потребителя да може да запаметява библиотеката и паролите на твърдия диск без да излиза от програмата.

7) Да се добавят потребителски имена

8) При търсене на книга да се извеждат няколко книги, които отговарят на критеритие за търсене. При други заявки, при които е необходима идентификиция чрез заглавие, да се даде възможност на потребителя да избере измежду книгите с еднакви заглавия.

Връзка към github: https://github.com/zahari02/OOP-Project1