**Глава 1**

Идеята на проекта е да се създаде електронна библиотека(подобно на „chitanкa”), в която книгите имат характеристики като автор, заглавие, рейтинг и т.н. Те трябва да могат да се местят и откриват посредством тези характеристики, трябва да могат да бъдат добавяни и премахвани в библиотеката и данните им да се записват в един общ файл, а съдържанието на всяка в отделен.

Задачите на разработката са да се създадат два класа(първият съответно за една книга , а вторият за цяла библиотека от такива) и наобходимите функции за изпълняване на идеята на проекта, като някои от функциите са с орторизиран достъп.

За да може да се работи с библиотеката, трябва да фигурират функции за добавяне и премахване на книга, като при премахването на книгата трябва да се изтрива и нейното съдържание(тоест файла, в който е записана). За да се реализира самият смисъл на понятието „библиотека” се създават функции, чрез които може да се открива една книга или по нейното заглавие, или по международния й стандартен номер, или по автора й, или и по описанието й. Като за описанието трябва да се има предвид, че може да се подаде низ, който е само част от цялото резюме, но програмата трябва да го разпознае. За откриването на книга по последния критерий трябва да се има предвид, че малък и голям символ, съотвестващ на дадена буква, трябва да се броят като равни. Освен това елементите на библиотеката трябва да могат да бъдат подреждани лексикографски спрямо авторите изаглавията, както и по големина на рейтинг, което изисква осъществяне на сортировки. Друга задача за изпълнение, за да бъде функционален проектът, е четенето от файл на съдържанието на книгата. То може да е до ред или до изречение, по избор на потребителя.

Целта на разработката е да осъществи изискваните задачи по правилен и с добра практика начин, чрез който да се постигне възможно най-добра функционалност на проекта.

Документацията по проекта ще бъде структурирана по глави, описващи основния идеята на проекта и идеите за осъществяването му. Ще представлява обобщение на смисъла на кода и свъразността му с изискваната функционалност на проекта.

**Глава 2**

Важно за кода е реализирането на голямата четврока, както и основни алгоритми като методът на пряката селекция за масив от указатели и цялостна работа с такъв масив. Важни и необходими са и алгоритмите за записване и четене в текстов файл. Основни дефиниции, които са необходими за осъществяването на проекта са тази на „канонично представяне”, на „сортиране”, на „четене от текстов файл”.

Поставената задача не е сложна като същност и принципи. Гради се на основните знания за работа с класове , масиви и файлове.

**Глава 3**

3.1:

Класовете са стандартни с private данни и private функции за копиране и триене на динамична памет, които след това се използват в public секцията. В нея е дефинирана и реализирана голямата четворка, както и други функции необходими за успешната реализация на кода. В проекта е създаден клас Book , в който е направена необходимата реализация за създаване на книга с нужните данни и функции. След него е създаден клас Library, в който се реализира масив от тип Book\*, което ни позволява да се възползваме от неговите функции и данни.

3.2:

За класа Book:

Голямата четворка:

//необходима е заради динамичната памет( char\* name, char\* autor, etc);

Book();//дефолтен конструктор

Book(const char\* autor, const char\* title, const char\* fileName, const char\* resume, unsigned rate, unsigned ibn);//конструктор с параметри

Book& operator=(Book const& other);//оператор =

Book(Book const& other);//копи-конструктор

~Book();

Реализация на записване в текстов файл:

void Book::writeToTextFileB(const char\* fileName) const {

//отваряме файла, в който засписваме книгите

std::ofstream out(fileName, std::ios::app);

if (!out) {//ако има проблем с отварянто функцията приключва

return;

}

out << \*this <<'+';//използваме предефиниран оператор << и разделител ‘+’

out.close();//затваряме файла

}

//при викане тази функция записва създадена книга във файл.

Функция, която чете до определен брой нови редове и отпечатва на конзолата прочетеното

void Book::readDscrFrTextFileLines(const char\* fileName, int numb) {

std::ifstream in(fileName);//oтваряме файла да четем от него

if (!in) {

return ;

}

int count = 0;//броят нови редове

while (true) { char buffer[1024];

in.getline(buffer, 1024);

std::cout << buffer; //отпечатваме реда

count++ ;//при край на getline(…) сме намерили нов ред и бройката се увеличава

if (count == numb || in.eof()) {

break;//цикълът се прекъсва, когато стигнем до зададената бройка или ако стигнем края на файла

}

}

in.close();//затваряме файла

}

За класа Library:

private:

Book\*\* books;//масив от указатели от тип Book

unsigned capacity;

unsigned currCt;

void clear();

void copy( Book\*\* books, unsigned capacity, unsigned currCt);

void resize();

public:

Голяма четворка:

Library();//дефолтен конструктор

Library( Book\*\* books, unsigned capacity, unsigned currCt);//конструктор с парамтери

Library& operator=(Library const& other);//оператор=

Library(Library const& other);//копиращ конструктор

~Library();//деструктор

Триене на книга:

void Library::deleteBook(const char\* title) {

for (int i = 0; i < currCt; i++) {

if (strcmp(books[i]->getTitle(), title) == 0) {//намира се книгата с подадено заглавие

books[i]->deleteFile(books[i]->getFileName());//трие се файлът с текста й

delete books[i];

books[i] = nullptr;//слага се да сочи към nullptr, защото в деструктора се трие указател и ако не се насочи ще сочи към чужда памет, което ще гърми

}

for (int j = i; j < currCt; j++) {

books[j] = books[j + 1];//тъй като е премахнат елемент от масива, трябва да се подреди

}

books[--currCt] = nullptr;

break;

}

}

Resize():

void Library::resize() {

Book\*\* buffer = new Book\*[capacity\*2];//заделя се нов масив с по-голям капацитет

for (int i = 0; i < capacity; i++) {

buffer[i] = books[i];//копират се всички указатели в него

}

capacity\*=2;

delete[]books;//трие се масиваът(но не и това, към което сочат указателите)

books = buffer;

}

Глава 4:

Реализация на класове:

-И за двата класа голямата четворка, private данните са основополагащи. Има реализирани деструктори за динамичната памет.По-важните фрагменти от кода са написани по-горе.

-Примери за тестване

Library books;

books.addBook("Dimityr dimov", "Tyutyun", "fileBook.txt", "tyutyunyt i grehyt", 10, 20, "fileArray.txt");

books.deleteBook("Tyutyun", "fileArray.txt");

...(съдържат се още примери в main())

Глава 5: Заключение

По - голямата част от началните цели се изпълняват от написания проект.