Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет   
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Дисциплина: Конструирование программ и языки программирования

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

на тему

ПРОГРАММА «МЕНЕДЖЕР ФАЙЛОВ-КНИГ»

БГУИР КП 1–400201.410 ПЗ

Студент: гр. 950504 Суша М. В.

Руководитель: старший преподаватель

Ковальчук А. М.

Минск 2020

Учреждение образования

«Белорусский государственный университет информатики   
и радиоэлектроники»

Факультет компьютерных систем и сетей

УТВЕРЖДАЮ

Заведующий кафедрой ЭВМ

\_\_\_\_\_\_\_\_\_\_Б. В. Никульшин

––"–––––––––––––––––2020г.

ЗАДАНИЕ

по курсовому проектированию

Студенту *Суше Матвею Викторовичу*

1. Тема проекта  Менеджер файлов-книг

2. Срок сдачи студентом законченного проекта *15 декабря 2020 г.*

*3. Исходные данные к проекту Файл: DataArtBooks – содержит информацию о художественных книгах, DataSciBooks – содержит информацию о научных книгах, DataNewspapers– содержит информацию о газетах, DataMAgazines – содержит информацию о журналах, . Контейнер: свой – tree и STL – stack.*

4. Содержание расчетно-пояснительной записки (перечень вопросов, которые подлежат разработке)

*Введение. Содержание. 1. Постановка задачи 2. Структура входных и выходных данных. 3. Диаграмма классов. 4. Описание классов. 5. Разработка алгоритмов. 5.1 Разработка алгоритмов по шагам. 5.2 Разработка cхем алгоритмов. 6. Тестирование. Заключение. Список литературы.*

5. Перечень графического материала (с точным обозначением обязательных чертежей и графиков)

*1. Диаграмма классов*

*2.Схема алгоритма template <class T>*\_iterator<T>& \_iterator<T>::operator++(*int*)

*3. Схема алгоритма template <class T>* node<T>\* tree<T>::\_add(T t, node<T>\* point, node<T>\* prepoint)

6. Консультант по проекту *Ковальчук А. М*.

7. Дата выдачи задания *12 сентября 2020 г.*

8. Календарный график работы над проектом на весь период проектирования (с обозначением сроков выполнения и трудоемкости отдельных этапов):

*разделы 1, 2 к 1 октября 2020 г. – 20 %;*

*разделы 3, 4 к 1 ноября 2020 г. – 30 %;*

*разделы 5, 6 ,7 к 1 декабря 2020 г. – 30 %;*

*оформление пояснительной записки и графического материала к 14 декабря 2020 г. – 20 %*

*Защита курсового проекта с 21 декабря 2020 г. по 28 декабря 2020 г.*

РУКОВОДИТЕЛЬ– –––––––– –*Ковальчук А. М.*

Задание принял к исполнению –Х *Суша М. В,*

**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ..........................................................................................................…5

1 ОБЗОР ИСТОЧНИКОВ.......................................................................................6

* 1. Постановка задачи........................................................................……............6
  2. Анализ аналогов программного средства......………………….……............6

2. СТРУКТУРНОЕ ПРОЕКТИРОВАНИЕ............................................................7

* 1. Структура программы………………...............................................................7
  2. Структура входных данных……………......…................................................7
  3. Структура промежуточных данных……......…...............................................9
  4. Структура выходных данных……......….........................………..................11

3. ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ................................................13

* 1. Диаграмма классов……………………………..............................................13
  2. Описание классов……………………....................………............................13

4. АЛГОРИТМЫ ..........................................................................................…..18

* 1. Алгоритмы по шагам.........................................................….........................18
  2. Схемы алгоритмов....................................................................................…...19

5 ТЕСТИРОВАНИЕ.......................................................................................…...20

ЗАКЛЮЧЕНИЕ................................................................................................….22

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ......................................…...23

ПРИЛОЖЕНИЕ А.................................................................................................24

ПРИЛОЖЕНИЕ Б..................................................................................................25

ПРИЛОЖЕНИЕ В..................................................................................................26

ПРИЛОЖЕНИЕ Г............................................................................................…..27

# **ВВЕДЕНИЕ**

# В современном мире стремительно развивающихся технологий книги не потеряли свою значимость, и литература перешла в свое новое электронное обличие. Я, как человек любящий читать, столкнулся с проблемой - за последние годы у меня скопилось огромное количество электронных книг, и теперь нет возможно найти нужную в этом беспорядке. Цель моего курсового проекта — навести порядок, создав менеджер электронных книг.

Для данной работы был выбран язык С++. C++ - компилируемый, статистический язык, придуманный Бьерном Страуструпом в 1983 году. С его помощью ежегодно создают сервера, прикладные программы разной сложности, компьютерные игры, встраиваемые системы, драйвера и многое другое. Он является мультипарадигменным, и данная работа написанна на основе объектоно-ориентированного и функцианалього парадигм. Существует большое количество реализаций данного языка, как например Microsoft Visual С++, Intel С++ compiler, Open64 C++ compiler и прочие. Я использовал GNU, так как он установлен по умолчанию в моей операционной системе (Ubuntu 20.04) .

Для реализации графического интерфейса я использовал Qt. Фреймворк - программное обеспечение, облегчающее разработку проекта. Он отличается от библиотеки тем, что не просто добавляет ряд функций, упрощающих работу, но сам диктует некоторые принципы архитектуры приложения. Для С++ существует ряд фреймоворков: GTK+, SFML, Cairo, Juce, wxWidgets и прочие. Но, как я писал выше, мой выбор пал на Qt. В начале работы над проектом я выбрал Qt, так как его IDE QtCreator явлется альтернативой Visual Studio в линукс. К концу проекта я выявил достоинства Qt. Его архитектура, построенная на системе сигнал-слот, хоть и имеет ряд ограничений, но все равно является очень удобным решением.

**1 ОБЗОР ИСТОЧНИКОВ**

**1.1 Постановка задачи**

Программа должна хранить информациию о книгах, журналах и газетах. Предоставлять возможность редактирования информации, сортировки, поиск. В файлах будут храниться: название, количество страниц, закладка, автор для книг, жанр для художественных книг, раздел науки для научных книг, территория для газет и тематика для журналов. Производится обработка исключительных ситуаций. Предусмотренна операция отмены последнего действия. Программа должна иметь удобный и понятный графический интерфейс. В программе будут реализованы функции:

- добавление, удаления и редактирования данных;

- сортировка данных по критериям;

- поиск по названию;

**1.2 Анализ аналогов программного средства**

Мной были найдены бесплатные аналоги планируемого приложения в репозиториях Ubuntu: Buka[1], Adobe Acrobat Reader[2], GNU Emacs[3], Raven Reader[4], coolreader3[5] и прочие. Все они совмещают в себе элементы менеджера книг и ридера. Их различие заключается в графическом интерфейсе и поддерживаемых файлах.

**2 СТРУКТУРНОЕ ПРОЕКТИРОВАНИЕ**

**2.1 Структура программы**

Приложение условно состоит из трех блоков: взаимодействия, анализа и хранения.

Блок взаимодействия(UI) — способ взаимодействия программы с пользователем. В данном блоке пользователь выбирает желаемое действие.

Блок анализа принимает информацию, введеную пользователем, и обрабатывает ее.

Из решений, принятых в блоке анализа, происходит работа с информацией в блоке хранения.

Структурная схема представлена на рисунке 2.1.1.

Рисунке 2.1.1. - Структурная схема

**2.2 Структура входных данных**

Информация в программе «Домашняя библиотека» будет храниться в бинарных файлах.

Структура данных, записываемых в файл, хранящий информацию о художественных книгах, представлена в таблице 2.2.1

Таблица 2.2.1 – Структура данных файла «ARTBooks»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| название | кол. страниц | закладка | автор | жанр |
| Дом, в котором... | 960 | 700 | Мариам  Петросян | роман |

Структура данных, записываемых в файл, хранящий информацию о научных книгах, представлена в таблице 2.2.2

Таблица 2.2.2 – Структура данных файла «SCBooks»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| название | кол. страниц | закладка | автор | раздел науки |
| Теория всего | 150 | 0 | Стивен Хокинг | физика |

Структура данных, записываемых в файл, хранящий информацию о газетах, представлена в таблице 2.2.3

Таблица 2.2.3 – Структура данных файла «Newspapers»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| название | кол. страниц | закладка | дата издания | Территория |
| The  Daily  Bugle | 20 | 10 | 5.12.20 | New-York |

Структура данных, записываемых в файл, хранящий информацию о журналах, представлена в таблице 2.2.4

Таблица 2.2.4 – Структура данных файла «Magazines»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| название | кол. страниц | закладка | дата издания | тема |
| Игромания | 25 | 0 | 3.12.20 | Компьютерные  игры |

**2.3 Структура промежуточных данных**

В качестве промежуточных данных использовались шаблон контейнера

бинарного дерева (tree), шаблон структуры Node, шаблон класса итератор, контейнер STL стек (stack).

***template*** *<****class*** *T>*

**class** **tree** **//** **Класс** **дерева**

{

**private**:

node<T> \*root;

*bool* (\*condition)(T, T);

**public**:

**typedef** \_iterator<T> iterator;

**tree**();

**tree**(T);

~**tree**();

*void* **add**(T); **//** **Добаление** **//B**

*void* **remove**(iterator); **//** **Удаление** **(дружественна** **итератору)**

*void* **remove**(node<T>\*); **//** **Удаление** **//B**

*void* **remove**(T); **//** **Удаление**

*void* **show**(); **//** **Вывод**

*void* **set\_condition** (*bool* (\*f) (T,T)); **//** **Задание** **нового** **условия**

tree<T>& **operator**= (tree<T>&);

\_iterator<T> **operator**[](*int*);

\_iterator<T> **begin**(); **//** **Для** **раоты** **с** **итератором**

\_iterator<T> **end**();

\_iterator<T> **rbegin**();

\_iterator<T> **rend**();

node<T>\* **find**(T t); **//** **Поиск**

\_iterator<T> **find2**(T t); **//** **Поиск** **возвращающий** **итератор**

**protected**:

node<T>\* **\_add**(T, node<T>\* pointer, node<T>\* prepointer = **nullptr**);

node<T>\* **\_find**(T t, node<T>\*);

**/\*** **Вспомогательные** **функции** **\*/**

\_iterator<T> **\_begin**(node<T>\*);

\_iterator<T> **\_rbegin**(node<T>\*);

*void* **\_copy**(node<T>\* point);

*void* **\_show**(node<T>\* point);

*void* **\_setroot**(node<T>\* p){root = p;}

**static** *bool* **\_condition**(T t1, T t2);

};

***template*** *<****class*** *D>*

**class** **\_iterator** **//** **Итератор**

{

**private**:

node<D> \*pointer;

**public**:

**\_iterator**(); **//** **Конструкторы**

**\_iterator**(node<D> \*p);

**\_iterator**(**const** \_iterator &it);

**/\*** **Перегруженные** **операторы\*/**

\_iterator &**operator**=(**const** \_iterator &it);

\_iterator &**operator**++(*int*);

\_iterator &**operator**++();

\_iterator &**operator**--(*int*);

D& **operator**\*();

*bool* **operator**!= (**const** \_iterator &it);

*bool* **operator**== (**const** \_iterator &it);

**friend** *void* tree<D>::**remove**(\_iterator<D>);

};

***template*** *<****class*** *T>*

**class** **node** **//** **Узел**

{

**friend** **class** tree<T>;

**friend** **class** \_iterator<T>;

**private**:

T data;

*unsigned* *int* height;

node<T> \*top; **//** **указатели** **на** **соседнии** **узлы**

node<T> \*left;

node<T> \*right;

T **getvalue**() **const** {**return** data;}; **//** **Взятие** **значения**

**public**:

**node**(T, node\* top = **nullptr**, node\* left = **nullptr**, **//** **Конструктор**

node\* right = **nullptr**,*unsigned* *int* hight = 1);

};

;

**2.4 Структура выходных данных**

Информация в программе «Домашняя библиотека» будет храниться в бинарных файлах.

Структура данных, записываемых в файл, хранящий информацию о художественных книгах, представлена в таблице 2.4.1

Таблица 2.4.1 – Структура данных файла «ARTBooks»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| название | кол. страниц | закладка | автор | жанр |
| Дом, в котором... | 960 | 700 | Мариам  Петросян | роман |

Структура данных, записываемых в файл, хранящий информацию о научных книгах, представлена в таблице 2.4.2

Структура данных, записываемых в файл, хранящий информацию о газетах, представлена в таблице 2.4.3

Структура данных, записываемых в файл, хранящий информацию о журналах, представлена в таблице 2.4.4

Таблица 2.4.2 – Структура данных файла «SCBooks»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| название | кол. страниц | закладка | автор | раздел науки |
| Теория всего | 150 | 0 | Стивен Хокинг | физика |

Таблица 2.4.3 – Структура данных файла «Newspapers»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| название | кол. страниц | закладка | дата издания | Территория |
| The  Daily  Bugle | 20 | 10 | 5.12.20 | New-York |

Таблица 2.4.4 – Структура данных файла «Magazines»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| название | кол. страниц | закладка | дата издания | тема |
| Игромания | 25 | 0 | 3.12.20 | Компьютерные  игры |

**3 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ**

**3.1 Диаграмма классов**

Диаграмма классов определяет типы классов системы и различного рода статические связи , которые существуют между ними.

Диаграмма классов данного проекта представлена в приложении А.

**3.2 Описание классов**

***class Reading // Кдасс литературы - книг***

{

**private**:

std::string name; **//** **Имя**

*unsigned* *int* pages; **//** **Количество** **страниц**

*unsigned* *int* top; **//** **закладка**

**public**:

**Reading**(); **//** **Конструкторы**

**Reading**(std::string name, *unsigned* *int* pages);

*void* **setName**(std::string); **//** **Сеттеры**

*void* **setPages**(*unsigned* *int*);

*void* **setTop**(*unsigned* *int*);

*void* **setTime**(QTime);

std::string **getName**(); **//** **Геттеры**

QDateTime **getTime**();

*unsigned* **getTop**();

*int* **getPages**();

*bool* **operator** > (**const** Reading&);**//** **Логические** **операторы**

*bool* **operator** < (**const** Reading&);

*bool* **operator**== (**const** Reading&);

**//** **Запись** **и** **чтение** **из** **файла**

**friend** std::ofstream& **operator**<< (std::ofstream &out, Reading &);

**friend** std::ifstream& **operator**>> (std::ifstream &in , Reading &);

**template** <**class** T>

**friend** *bool* **conditionPagesIncr**(T t1, T t2); **//** **условия** **для** **дерева**

**template** <**class** T>

**friend** *bool* **conditionPagesDecr**(T t1, T t2);

};

***class*** *Books :* ***public*** *Reading* ***//******Книги***

{

**private**:

std::string author; **//** **Автор**

**public**:

**Books**(); **//** **Конструкторы**

**Books**(std::string name, *unsigned* *int* pages,

std::string author);

std::string **getAuth**(); **//** **Геттеры**

*void* **setAuth**(std::string); **//** **Сеттры**

*bool* **operator** > (**const** Books&); **//** **Логические** **операторы**

*bool* **operator** < (**const** Books&);

*bool* **operator**== (**const** Books&);

**friend** std::ofstream& **operator**<< (std::ofstream &out, Books &);

**//** **Запись**

**friend** std::ifstream& **operator**>> (std::ifstream &in , Books &);

**//** **и** **чтение** **из** **файла**

};

***class******ARTBooks*** *:* ***public*** *Books* ***//******Художественные******книги***

{

**private**:

std::string genre; **//** **Жанр**

**public**:

**ARTBooks**(); **//** **Конструкторы**

**ARTBooks**(std::string name, *unsigned* *int* pages = 0,

std::string author = "unknown", std::string genre = "unknown");

std::string **getGenre**(); **//** **Геттер**

*void* **setGenre**(std::string); **//** **Сеттер**

*bool* **operator** > (**const** ARTBooks&);**//** **Логичесие** **операторы**

*bool* **operator** < (**const** ARTBooks&);

*bool* **operator**== (**const** ARTBooks&);

**friend** std::ofstream& **operator**<< (std::ofstream &out, ARTBooks &);

**//** **Запись**

**friend** std::ifstream& **operator**>> (std::ifstream &in , ARTBooks &);

**//** **и** **чтение** **из** **файла**

};

***class*** *SCBooks :* ***public*** *Books* ***//******Научные******книги***

{

**private**:

std::string section; **//** **Раздел** **науки**

**public**:

**SCBooks**(); **//** **Конструкторы**

**SCBooks**(std::string name, *unsigned* *int* pages = 0,

std::string author = "unknown", std::string section = "unknown");

std::string **getSection**(); **//** **Геттер**

*void* **setSection**(std::string); **//** **Сеттер**

*bool* **operator** > (**const** SCBooks&); **//** **Логический** **оператор**

*bool* **operator** < (**const** SCBooks&);

*bool* **operator**== (**const** SCBooks&);

**friend** std::ofstream& **operator**<< (std::ofstream &out, SCBooks &);

**//** **Запись**

**friend** std::ifstream& **operator**>> (std::ifstream &in , SCBooks &);

**//** **и** **чтение** **из** **файла**

};

***class******TempEdition*** *:* ***public*** *Reading* ***//******Перодическеие******издания***

{

**private**:

QDateTime date\_of\_issue; **//** **Дата** **издания**

**public**:

**TempEdition**(); **//** **Конструкторы**

**TempEdition**(std::string name, *unsigned* *int* pages);

QDateTime **getTimeIssue**(); **//** **Геттер**

*void* **setTimeIssue**(QDateTime); **//** **Сеттер**

*bool* **operator** > (**const** TempEdition&); **//** **Логические** **операторы**

*bool* **operator** < (**const** TempEdition&);

*bool* **operator**== (**const** TempEdition&);

**friend** std::ofstream& **operator**<< (std::ofstream &out, TempEdition &);

**//** **Запись**

**friend** std::ifstream& **operator**>> (std::ifstream &in , TempEdition &);

**//** **и** **чтение** **из** **файла**

};

**class** Magazines : **public** TempEdition **//** **Журналы**

{

std::string subject; **//** **Тема**

**public**:

**Magazines**(); **//** **Конструкторы**

**Magazines**(std::string name, *unsigned* *int* pages = 0, std::string subjects = "unknown");

std::string **getSubject**(); **//** **Геттер**

*void* **setSubject**(std::string); **//** **Сеттер**

*bool* **operator** > (**const** Magazines&); **//** **Логические** **операторы**

*bool* **operator** < (**const** Magazines&);

*bool* **operator**== (**const** Magazines&);

**friend** std::ofstream& **operator**<< (std::ofstream &out, Magazines &);

**//** **Запись**

**friend** std::ifstream& **operator**>> (std::ifstream &in , Magazines &);

**//** **и** **чтение** **из** **файла**

};

**class Newspapers : public TempEdition // Газета**

{

**private**:

std::string location; **//** **Местоположение**

**public**:

**Newspapers**(); **//** **КОнструктор**

**Newspapers**(std::string name, *unsigned* *int* pages = 0, std::string location = "unknown");

std::string **getLocation**(); **//** **Геттер**

*void* **setLocation**(std::string); **//** **Сеттер**

*bool* **operator** > (**const** Newspapers&);**//** **Логические** **операторы**

*bool* **operator** < (**const** Newspapers&);

*bool* **operator**== (**const** Newspapers&);

**friend** std::ofstream& **operator**<< (std::ofstream &out, Newspapers &);

**//** **Запись**

**friend** std::ifstream& **operator**>> (std::ifstream &in , Newspapers &);

**//** **и** **чтение** **из** **файла**

};

**4 РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ**

**4.1 Алгоритмы по шагам**

Алгоритм по шагам метода

template <class T> node<T>\* tree<T>::\_find(T t, node<T>\* point)

1. Начало.

2. Входные данные

T – тип, передаваемый шаблону;

t — искомый эллемент типа T;

point — указатель на текущий узел;

Выходные данные:

point — указатель на узел искомого эллемента.

3. Если point равен nullptr, то переходим к шагу 7, иначе — шаг 4.

4. Если point→ data равна t, то возвращаем point и переходим к шагу 8, иначе — шаг 5.  
 5. Если condition(t, point) равен true возвращаем результат функции \_find(t, point→right) и переходим к шагу 8, иначе — шаг 6.

6. Если condition(t, point) равен false возвращаем результат функции \_find(t, point→left) и переходим к шагу 8 , иначе — шаг 7.

7. Возвращаем nullptr.

8. Конец.

Алгоритм по шагам метода

template <class T> \_iterator<T> tree<T>::operator[](*int* num)

1. Начало.

2. Входные данные

T – тип, передаваемый шаблону;

num — порядковый номер искомого элемента;

i — переменная типа int, порядковое значение элементов;

it — итератор по дереву;

Выходные данные:

it — итератор искомого элемента.

3. Если num меньше 0 возвращаем nullptr и шаг - 10, иначе шаг 4.

4. Инициализируем i нулем.

5. Цикл по контейнеру tree.

6. Если i равен num возвращаем it и шаг - 10, иначе шаг - 7

7 i++.

8. Конец цикла по контейнеру tree.

9. Возвращаем итератор end, указывающиий на конец контейнера tree.

10. Конец.

**4.2 Схемы алгоритмов**

Схема алгоритма шаблона функции template <class T>

\_iterator<T>& \_iterator<T>::operator++(*int*) приведена в приложении Б.

Схема алгоритма метода шаблона класса template template <class T>

node<T>\* tree<T>::\_add(T t, node<T>\*, node<T>\*) приведена в приложении В.

5.**ТЕСТИРОВАНИЕ**

На этапе тестирование проводится проверка работы приложения. Запустим приложение (рисунок 5.1)

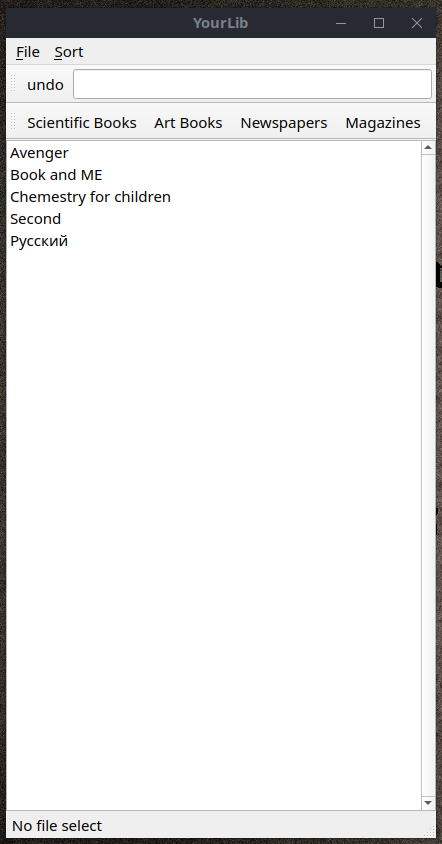


Рисунок 5.1

По очередно проверим работу кнопок «Scientific Books», «Art Books», «Newspapers», «Magazines»(рисунок 5.2).

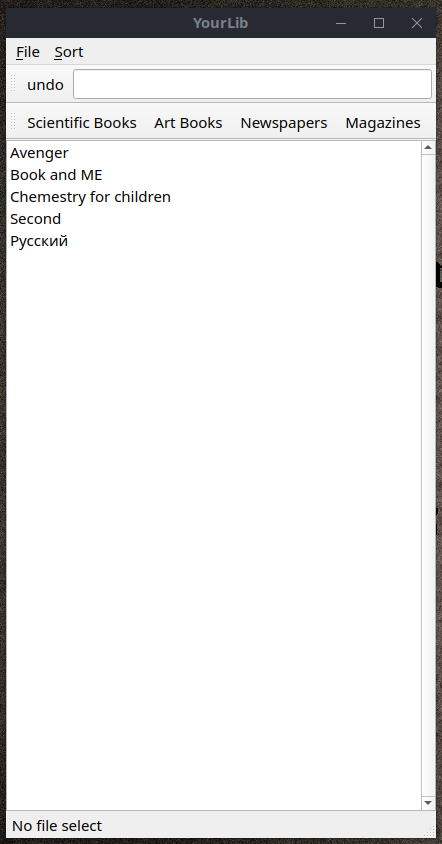
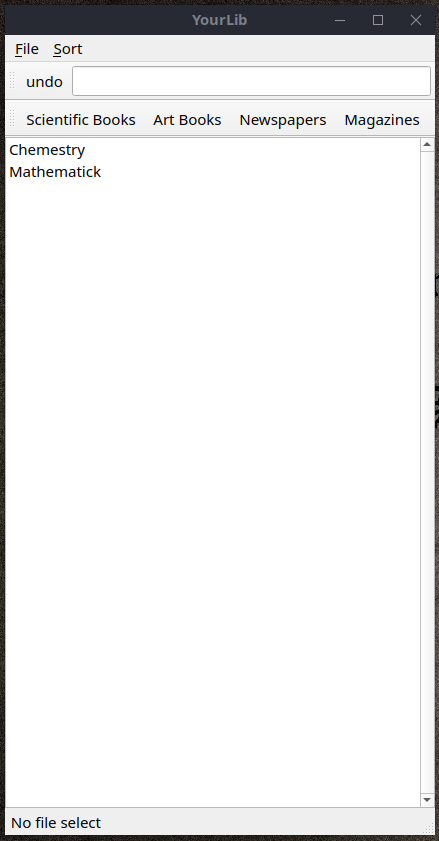
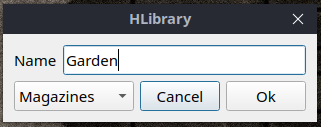
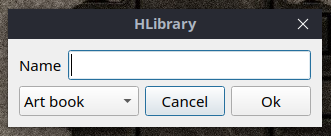
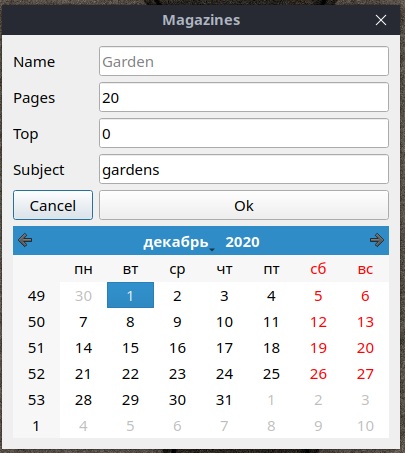
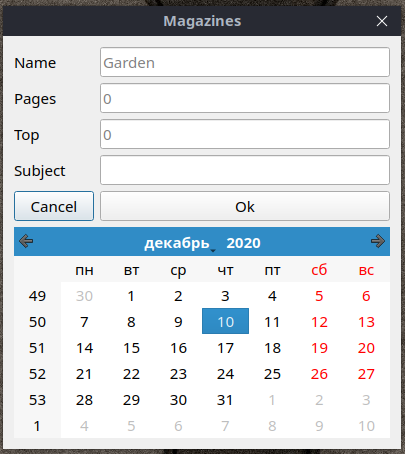


Рисунок 5.2

Попробуем добавить Журнал (рисунок 5.3)

 Рисунок 5.3

Добавим информацию о журнале (рисунок 5.4)



**ЗАКЛЮЧЕНИЕ**

В ходе выполнения данного курсового проекта были применено на практике знания, полученные самостоятельно, а так же в ходе изучения курса КпиЯП. Была реализованна программа «Домашняя библиотека». Задание было выполнено и протестировано.

Приложение может быть усовершенствованно:

- добавлением системы ридера;

- создание резервной копии данных в облаке;

- создание системы тэгов и построение на их основе персональных предложений пользователю.

**ЛИТЕРАТУРА**

[1] «Buka» — приложение для управления электронными книгами [Электронный ресурс]. – Режим доступа: https://www.electronjs.org/apps/buka/.

# [2] «Adobe Acrobat Reader» — приложение для просмотра, печати и комментирования документов в формате PDF[Электронный ресурс]. – Режим доступа: <https://get.adobe.com/ru/reader/otherversions/>.

# [3]«GNU Emacs» — многофункциональный текстовый редактор[Электронный ресурс]. – Режим доступа: <https://www.gnu.org/software/emacs/>.

[4]«Raven Reader» — программа для чтения новостей и статей [Электронный ресурс]. – Режим доступа: https://ravenreader.app/.

# [5]«CoolReader 3» — приложение для чтения электронных книг [Электронный ресурс]. – Режим доступа: [https://www.fb2-reader.ru/read/cool\_reader\_3/](https://health-diet.ru/health_diet/).