

Колледж космического машиностроения и технологий

КУРСОВОЙ ПРОЕКТ

По МДК.02.02 «Технологии разработки и защиты баз данных»

Тема: «Разработка автоматизированного рабочего места библиотекаря»

Выполнил студент

Зайцев Никита Евгеньевич

Группа П2-18

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(Подпись)

\_\_\_\_\_\_\_\_\_\_\_\_ (Дата сдачи работы)

Проверил преподаватель

Попов Вячеслав Николаевич

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(Подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(Оценка)

Королев 2022 г.

Оглавление

[**Введение** 3](#_Toc105785327)

[**1.** **Теоретическая часть** 4](#_Toc105785328)

[**1.1.** **Описание предметной области** 4](#_Toc105785329)

[**1.2.** **Описание существующих разработок** 6](#_Toc105785330)

[**1.2.1.** **HelloClient** 6](#_Toc105785331)

[**1.2.2.** **1С:Библиотека** 7](#_Toc105785332)

[**2.** **Проектная часть** 9](#_Toc105785333)

[**2.1.** **Диаграмма прецедентов** 9](#_Toc105785334)

[**2.2.** **Выбор инструментов** 10](#_Toc105785335)

[**2.3.** **Проектирование сценария** 11](#_Toc105785336)

[**2.5.** **Описание главного модуля** 13](#_Toc105785337)

[**2.6.** **Описание спецификаций к модулям** 16](#_Toc105785338)

[**2.7.** **Описание модулей** 17](#_Toc105785339)

[**2.9. Описание применения средств отладки.** 21](#_Toc105785340)

[**3.** **Эксплуатационная часть** 22](#_Toc105785341)

[**3.1.** **Руководство пользователя** 22](#_Toc105785342)

[**3.2 Элементы управления** 27](#_Toc105785343)

[**Заключение** 28](#_Toc105785344)

[**Список литературы и интернет-источников** 29](#_Toc105785345)

[**Приложения** 30](#_Toc105785346)

# **Введение**

Целью данного курсового проекта является написание программы «Library» для автоматизации рабочего места библиотекаря.

Такая программ облегчит поставку книг. Простой интерфейс для использования базы данных дает возможность быстро понять управление в программе.

Автоматизация любой библиотеки - это, в первую очередь, повышение производительности и эффективности труда, улучшение качества информационных услуг, устранение трудоемких и монотонных операций.

В первой части будет рассмотрена предметная область данной темы, а также несколько продуктов по данной теме.

Во второй части будут рассмотрены инструменты и модули, которые были разработаны, структура программной части и листинги ключевых частей программных модулей.

В третьей части будет рассмотрено руководство для потенциальных пользователей.

В заключительной части будет приведен To-do лист с планами по доработки программы, а также сделаны общие выводы о получившемся проекте.

# **Теоретическая часть**

## **Описание предметной области**

Библиотека - это учреждение культуры, организующее сбор, хранение и общественное пользование произведениями печати и другими документами. Библиотеки систематически занимаются сбором, хранением, пропагандой и выдачей читателям произведений печати, а также информационно-библиографической работой, являются общедоступным источником знаний и основной базой для самообразования.

Основными направлениями работы любой библиотеки являются: комплектование и организация книжного фонда; обслуживание читателей.

Комплектование фондов библиотеки состоит из систематического выявления (путем просмотра библиографических источников и литературы) нужных для данной библиотеки изданий и приобретения их. От своевременности и полноты комплектования библиотеки в значительной мере зависит уровень обслуживания читателей.

Организация книжного фонда включает вопросы учета, расстановки, хранения литературы и доставки ее читателю. Правильная организация фонда облегчает читателю пользование литературой, библиотекарю - быстрое выполнение читательских требований, а также обеспечивает сохранность фондов как общественной собственности.

Обслуживание читателей библиотеки осуществляется различными путями:

- выдача литературы, как в читальный зал, так и за пределы библиотеки;

- помощь отдельным читателям и учреждениям в подборе необходимой им литературы;

- раскрытие книжных фондов библиотеки через систему библиотечных каталогов;

- составление информационно-библиографических пособий различного типа;

- пропаганда наиболее ценной литературы;

- репродуцирование текстов по заказам читателей и т.д.

Библиотечная деятельность связана с учетом большого количества операций, множество книг и читателей серьезно замедляют работу библиотекарей. Сложность поиска нужной книги в каталоге занимает длительное время, и целиком опирается на компетентность работников библиотеки.

В библиотеке ведется картотека читателей. Для ведения библиотечных каталогов, организации поиска требуемых изданий и библиотечной статистики в базе должны храниться сведения, большая часть которых размещаются в аннотированных каталожных карточках

Каждой библиотеке нужно грамотное планирование поставок книг и в этом моя программа будет незаменима.

Раньше все книги записывались вручную на бумагу и такую документацию очень сложно было вести ведь она могла потеряться или испортиться от физических воздействий. Сейчас же все пишется на компьютере и сохраняется на серверах в надежном месте.

В данной программе можно полностью манипулировать всеми данными книг. Записывать новые, изменять или дополнять и конечно же сохранять для дальнейшего использования.

Для правильного составления поставок книг требуется вся необходимая информация, начиная от типа книги и заканчивая ее годом выпуска. Все начинается с определения типа книги, ведь допустим более старый год выпуска будет возможно продать за намного большую стоимость.

## **Описание существующих разработок**

В этом разделе рассмотрены некоторые уже имеющиеся программы для просмотра поставок книг, и также дополнительной информации о годе выпуска и стоимости.

Интерфейс таких программ и сайтов для узнавания информации о книгах чаще всего простой и понятный для обычных пользователей.

### [**HelloClient**](https://www.livebusiness.ru/tool/2752/)

Сервис для управления поставкой книг и любым другим товаром. Возможность создать единую базу клиентов с историей всех обращений и покупок, а также возможность добавлять товары и принимать оплату.

Одна из самых популярных программ для компаний который предоставляют какие-либо услуги, в интернете есть множество положительных отзывов о данном инструменте.

Достоинства, которые подтверждают отзывы:

1. Легкодоступная.
2. Приятный интерфейс и дизайн, который хорошо гармонирует между собой.
3. Соответствует самым требовательным функциям.
4. Гарантирует сохранность всей информации о товаре.

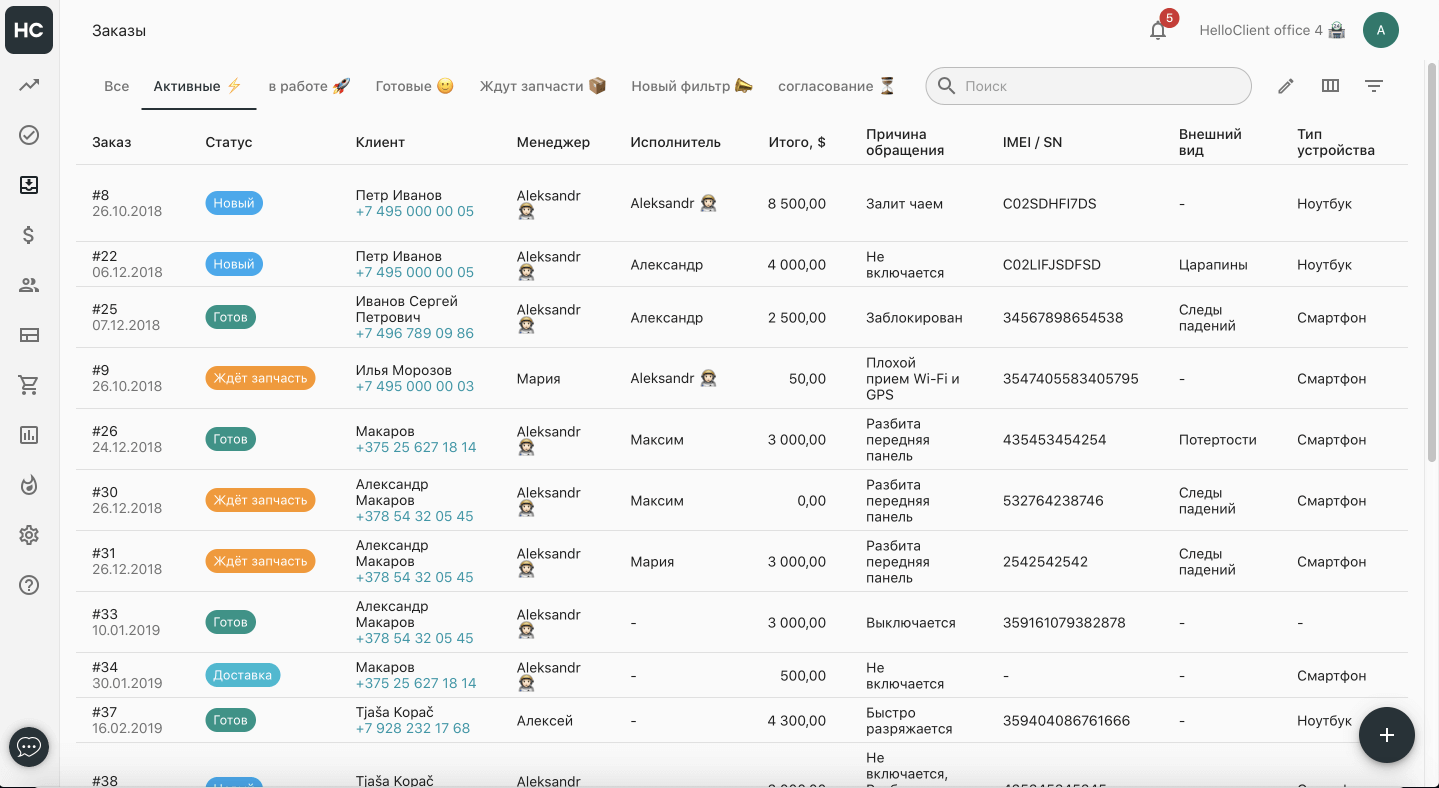


Рисунок 4. Интерфейс [“HelloClient”](https://www.livebusiness.ru/tool/2752/)

### **1С:Библиотека**

Программа для автоматизации деятельности библиотек любого типа и назначения. Продукт позволяет автоматизировать рабочие процессы библиотеки, в зависимости от ее назначения, типа, состава фондов, может быть интегрирован с другими типовыми решениями фирмы "1С". Есть конфигурации для школы, ВУЗа, колледжа.

Особенности:

1. Вся информация так же доступна работнику и при необходимости он может отредактировать или удалить определенную книгу.
2. Заслуженное доверие многих компаний за время своего существования.
3. Имеет минималистический дизайн приятный глазу.

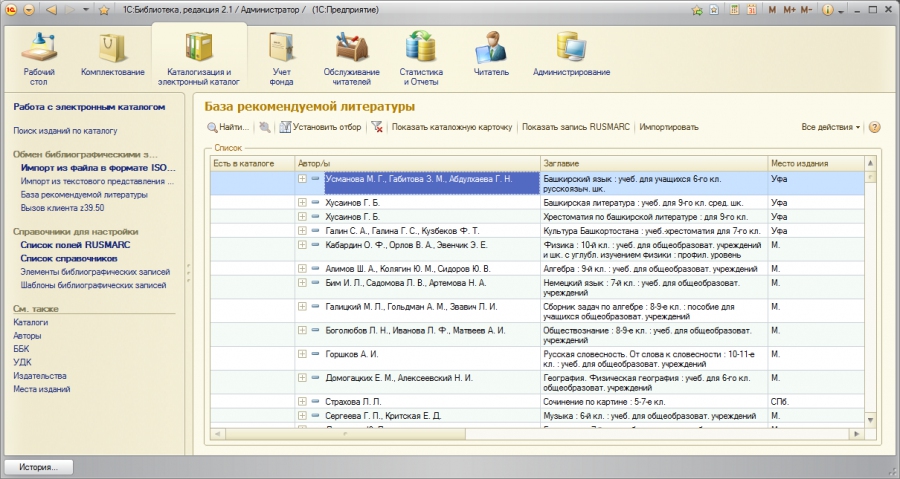


Рисунок 5. Интерфейс “1С:Библиотека”

# **Проектная часть**

## **Диаграмма прецедентов**

В этом разделе представлены три диаграммы прецедентов. На них показаны все возможные функциональные и поведенческие отношения.

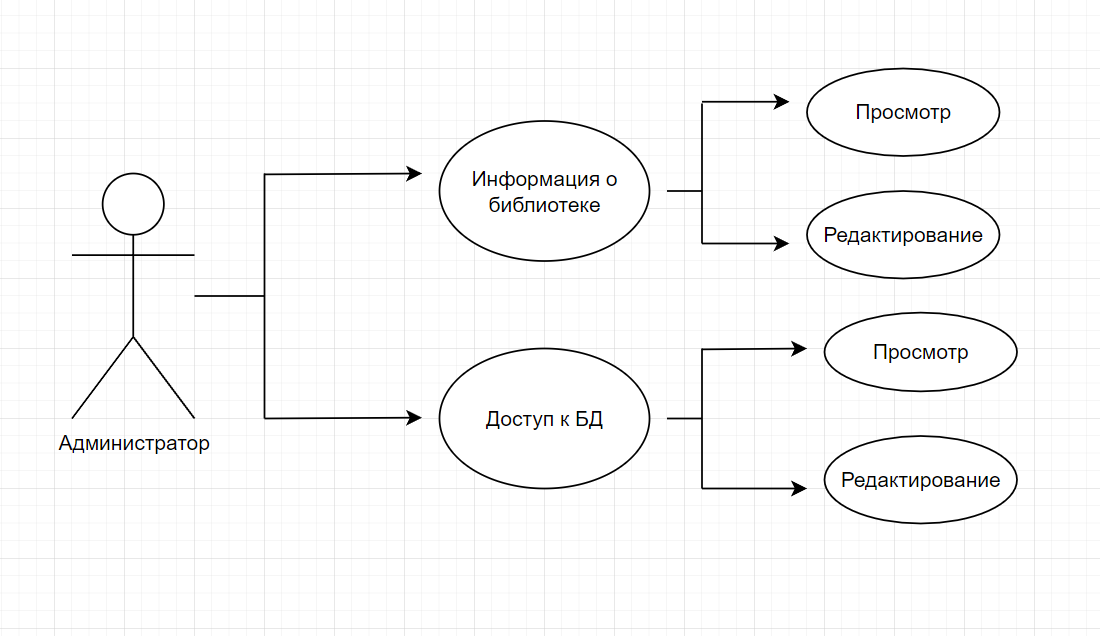
**

Рисунок 6. Диаграмма прецедентов –Администратор

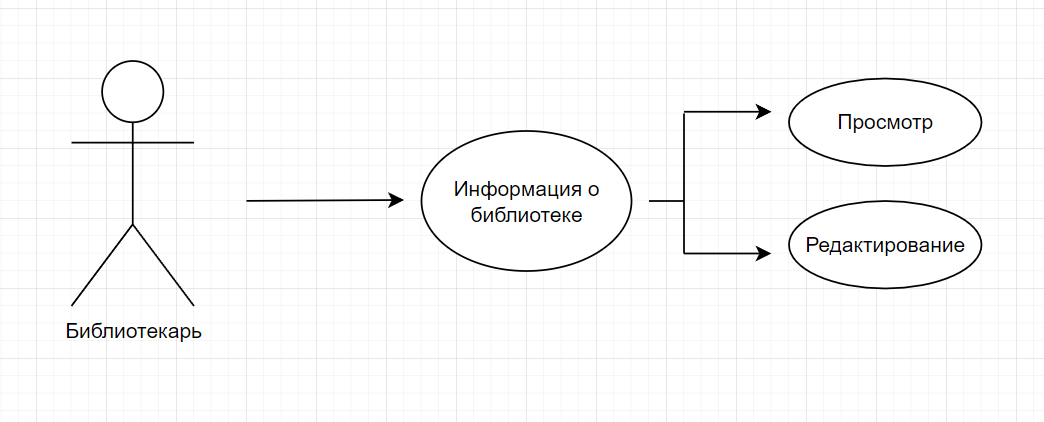


Рисунок 7. Диаграмма прецедентов – Библиотекарь

## **Выбор инструментов**

При выборе инструментов было проведено сравнение по критериям, представленных в таблице 1.

Степень важности критерия выбиралась из: низкая, ниже средней, средняя, ниже высокой, высокая.

Таблица 1.

Критерии выбора инструмента.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Критерий | Участие в корпоративном проекте | Простота сопровождения | Наличие библиотек | Наличие документации на русском языке | Скорость разработки |
| Важность критерия | Низкая | Средняя | Высокая | Ниже средней | Ниже высокой |

Исходя из этих критериев, я сравнил 3 языка программирования от 0 до 10 баллов за критерий.

Таблица 2.

Оценка языков программирования.

|  |  |  |  |
| --- | --- | --- | --- |
|  | C++ | Python | Object Pascal |
| Участие в корпоративном проекте | 10 | 8 | 4 |
| Простота сопровождения | 7 | 10 | 3 |
| Наличие библиотек | 6 | 10 | 4 |
| Наличие документации на русском языке | 8 | 6 | 6 |
| Скорость разработки | 6 | 10 | 3 |
| Итого баллов | 37 | 44 | 20 |

По результатам сравнения был выбран язык программирования Python.

## **Проектирование сценария**

В данном разделе приведен сценарий использования программы обычным пользователем.

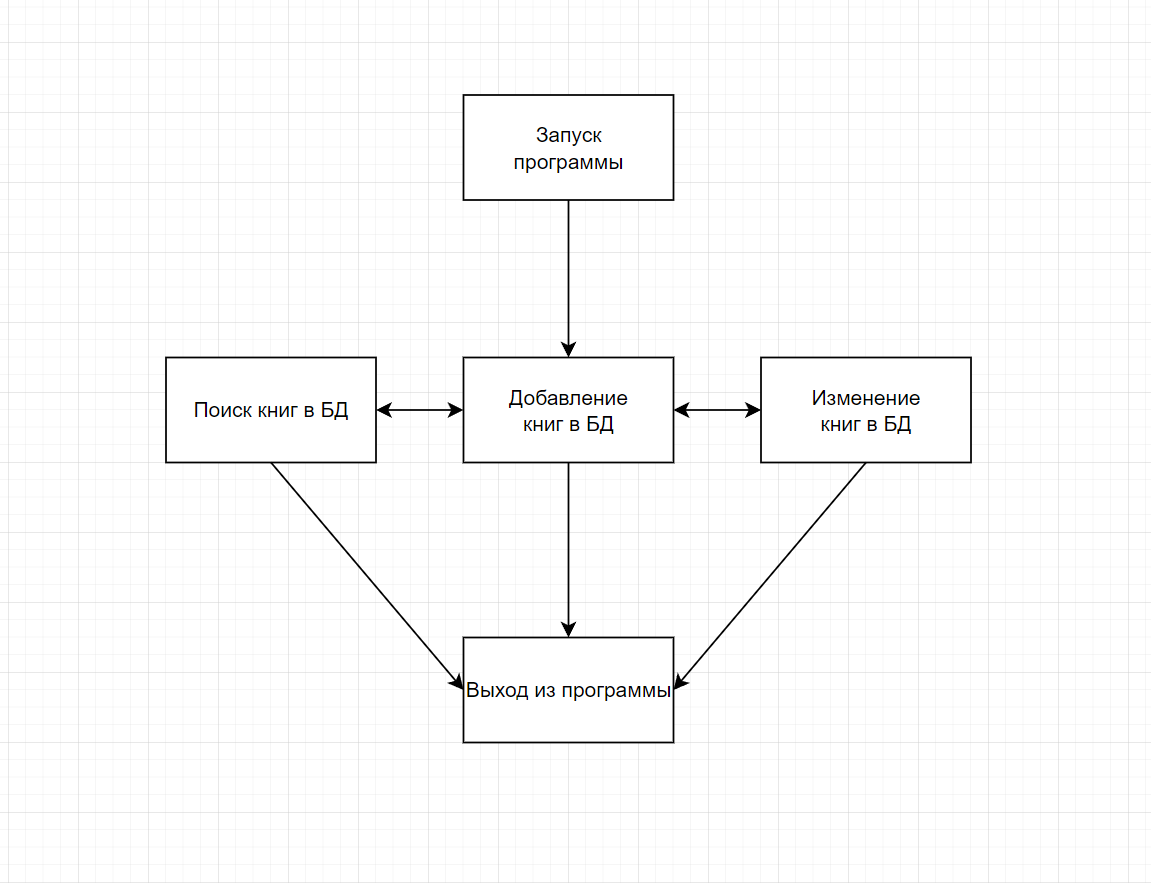


Рисунок 9. Сценарий использования

Пользователь после запуска программы может выполнить 4 действия: добавить книгу, отыскать книгу, редактировать книгу и непосредственно выйти из программы.

При выборе выхода, программа заканчивает свою работу, после произведения манипуляций с информацией о книгах, программа переходит в режим ожидания.

* 1. **Модуль базы данных**

При разработке приложения “Автоматизированное рабочее место библиотекаря” было принято решение использовать нереляционный тип базы данных. Так как данный тип баз данных проще для добавления, изменения и поиска входных данных многоструктурных и сложно кодируемых данных.

Нереляционная база данных — это база данных, в которой в отличие от большинства традиционных систем баз данных не используется табличная схема строк и столбцов. В этих базах данных применяется модель хранения, оптимизированная под конкретные требования типа хранимых данных. Например, данные могут храниться как простые пары "ключ — значение", документы JSON или граф, состоящий из ребер и вершин.

Все эти хранилища данных не используют реляционную модель. Кроме того, они, как правило, поддерживают определенные типы данных. Процесс запроса данных также специфический. Например, хранилища данных временных рядов рассчитаны на запросы к последовательностям данных, упорядоченных по времени. В свою очередь хранилища данных графов рассчитаны на анализ взвешенных связей между сущностями. Ни один из форматов не подходит в полней мере при выполнении задач управления данными о транзакциях.

## **Описание главного модуля**

Главный модуль называется “ LibTK.py”, он же является точкой запуска. В него подключаются набор библиотек:

1. Tkinter
2. Tkinter, ttk
3. SqLite3

**Листинг 1. Сохранение информации в базу данных.**

def savedBD():

conn = sqlite3.connect('LibTable')

curs = conn.cursor()

curs.execute("DELETE FROM Lib")

a = []

for l, i in enumerate(mainlist.get(0, END)):

a.append([])

for j in i.split(' '):

a[l].append(j)

curs.executemany("INSERT INTO Lib VALUES (?, ?, ?, ?)", a)

conn.commit()

Данный метод сохраняет все внесенные изменения в базу данных.

**Листинг 2. Обновление отображения**

def update():

conn = sqlite3.connect('LibTable')

curs = conn.cursor()

\*grid, = curs.execute("""SELECT \*

FROM Lib""")

mainlist.delete(0, END)

for i in grid:

mainlist.insert(END, str(i[0]) + ' ' + str(i[1]) + ' ' + \

str(i[2]) + ' ' + str(i[3]))

Данный метод обновляет отображение содержимого базы данных в специальном поле просмотра.

**Листинг 3. Добавление книги в БД**

def add():

mainlist.insert(END, NameVar1.get() + ' ' \

+ AuthorVar2.get() + ' ' \

+ YearVar3.get() + ' ' \

+ PriceVar4.get())

NameVar1.set('')

AuthorVar2.set('')

YearVar3.set('')

PriceVar4.set('')

savedBD()

Данный метод добавляет в базу данных новые данные о книгах (данный метод не сохраняет, а только добавляет данные).

**Листинг 4. Поиск данных о книге из БД**

def getk():

global rowtext, mainlist, Yearbox1, Yearbox2

conn = sqlite3.connect('LibTable')

curs = conn.cursor()

\*gridAuto, = curs.execute("""SELECT \*

FROM Lib

WHERE Year BETWEEN ? AND ?

ORDER BY Year DESC""",\

(Yearbox1.get(), Yearbox2.get()))

mainlist.delete(0, END)

for i in gridAuto:

\*i, = i

rowtext.set(str(i[0]) + ' ' + str(i[1]) + '' + \

str(i[2]) + ' ' + str(i[3]))

mainlist.insert(END, rowtext.get())

Данный метод начинает поиск информации из базы данных.

**Листинг 5. Изменение информации о книгах в БД**

def change():

select = list(mainlist.curselection())

mainlist.delete(select)

mainlist.insert(select, NameVar1.get() + ' ' \

+ AuthorVar2.get() + ' ' \

+ YearVar3.get() + ' ' \

+ PriceVar4.get())

NameVar1.set('')

AuthorVar2.set('')

YearVar3.set('')

PriceVar4.set('')

savedBD()

Данный метод изменяет частично или полностью информацию о книге (опять же этот метод не сохраняет данные изменения).

## **Описание спецификаций к модулям**

В данном разделе описаны публичный интерфейс модулей курсового проекта.

В главном модуле (LibTK.py) содержатся публичные методы функций getk, saveBD, update, add, change которые наследуется от функции main. Так как все эти методы были описаны в разделе 2.4, в данном разделе описание данных методов опущено. В данном модуле также есть несколько объектов:

1. tkinter – объект, который создает системное окно и объект самого приложения.
2. Sqlite3 – объект для взаимодействия с базой данных.
3. LibData.py – объект который который создает базу данных.

Модуль, который создает базу данных (LibData.py) будет описан ниже в разделе 2.7.

## **Описание модулей**

Кроме главного модуля программа содержит еще один модуль который создает базу данных (LibData.py).

В этом разделе приведены все методы, которые содержатся в модуле LibData.py.

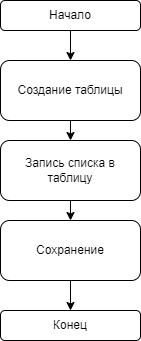


Рисунок 10. Блок-схема модуля LibData.py

* 1. **Описание тестовых наборов модулей**

В этом разделе будет визуализирован процесс работы приложения в виде тестов:

Тест 1. Запуск приложения.

Действия: Запустить exe файл.

Ожидаемый результат: Успешный запуск окна.

Результат теста:



Рисунок 11. Запуск программы

Тест 2. Поиск книг.

Действия: написать любой текст или число в одну из вкладок слева.

Ожидаемы результат: Поиск по введенному тексту или числу.

Результат тесте:



Рисунок 12. Поиск книг

Тест 3. Изменение информации о книге.

Действия: Выбрать нужную книгу, затем заполнить поля новыми данными и нажать кнопку “Изменить”.

Ожидаемый результат: Изменение информации о книге.

Результат теста:

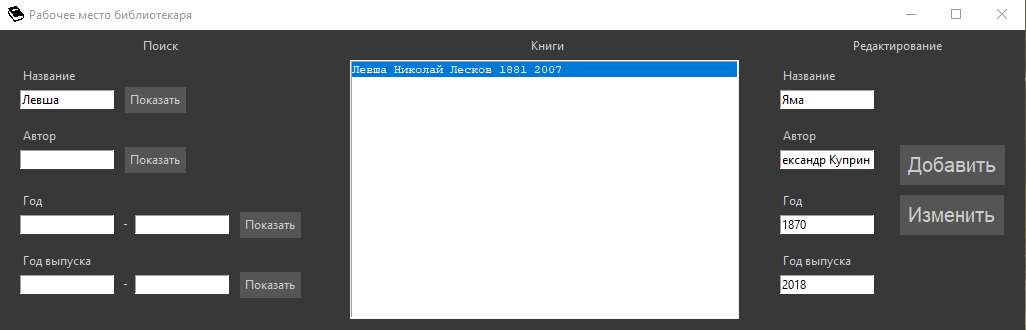


Рисунок 13. Заполнение полей

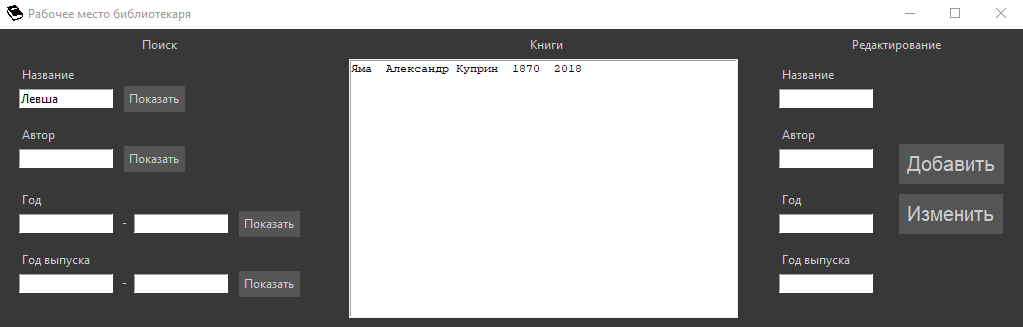


Рисунок 14. Применение новых значений

Тест 4. Добавление новой книги

Действия: Заполнить поля нужными данными о книге и нажать кнопку “Добавить”.

Ожидаемый результат: Добавление новой книги.

Результат теста:

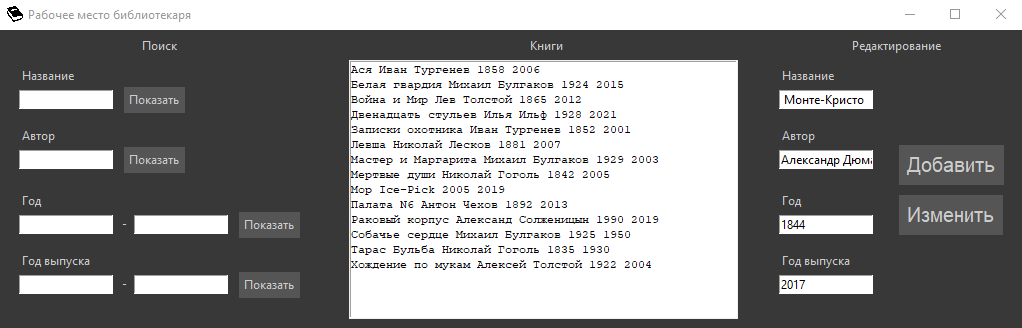


Рисунок 15. Заполнение полей

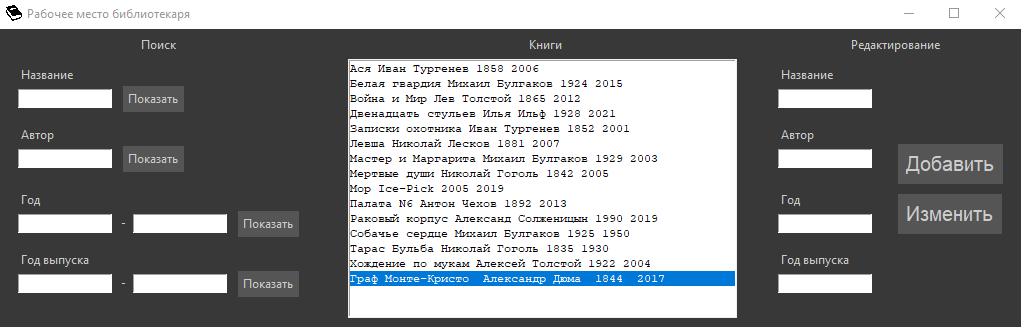
**

Рисунок 16. Результат добавления книги

## **2.9. Описание применения средств отладки.**

Для Грамотной отладки в Visual Studio Code предусмотрен Debug режим который не только позволяет наглядно узнать, что где находится, но еще и строит иерархию зависимостей и наследования.

Ниже представлен пример данных из отладчика Visual Studio Code

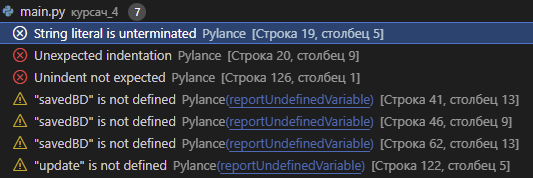


Рисунок 17. Пример Запущенного отладчика

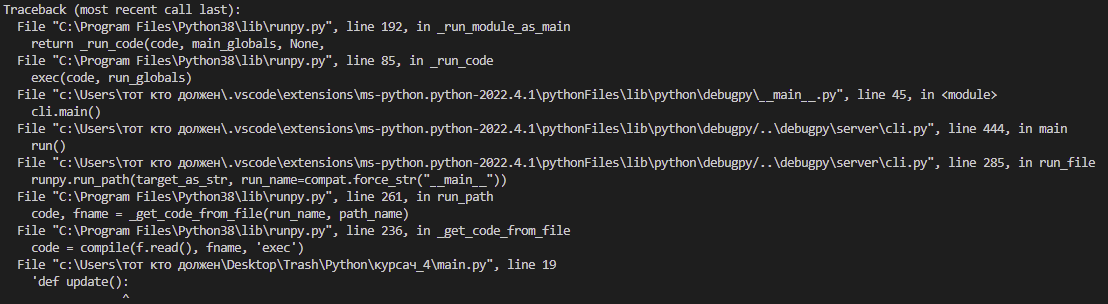


Рисунок 18. Пример выведенных ошибок

С помощью такого подхода можно отловить не только те баги которые связаны с неточностями во время написания синтаксиса и явно себя проявляющие на этапе запуска, по типу “ValueError”, но и определить пример неправильного наследования или утечку памяти.

# **Эксплуатационная часть**

## **Руководство пользователя**

**АННОТАЦИЯ**

В данном программном документе приведено руководство пользователя по применению и эксплуатации программы «Library», предназначенной для автоматизации рабочего места библиотекаря.

В данном программном документе, в разделе «Назначение программы» указаны сведения о назначении программы и информация, достаточная для понимания функций программы и ее эксплуатации.

В разделе «Условия выполнения программы» указаны условия, необходимые для выполнения программы (минимальный состав аппаратных и программных средств и т.п.).

В данном программном документе, в разделе «Выполнение программы» указана последовательность действий оператора, обеспечивающих загрузку, запуск, выполнение и завершение программы, приведено описание функций, формата и возможных вариантов команд, с помощью которых пользователь управляет выполнением программы, а также ответы программы на эти команды.

Оформление программного документа «Руководство оператора» произведено по требованиям ЕСПД (ГОСТ 19.101-77 [[1]](#footnote-1)1), ГОСТ 19.103-77 [[2]](#footnote-2)2), ГОСТ 19.104-78\* [[3]](#footnote-3)3), ГОСТ 19.105-78\* [[4]](#footnote-4)4), ГОСТ 19.106-78\* [[5]](#footnote-5)5), ГОСТ 19.505-79\* [[6]](#footnote-6)6), ГОСТ 19.604-78\* [[7]](#footnote-7)7)).

1. **Назначение программы**
   1. **Функциональное назначение программы**

Специальное программное обеспечение «Library» используется для составления информации о книгах.

* 1. **Эксплуатационное назначение программы**

Специальное программное обеспечение «Library» может эксплуатироваться специализированным пользователем компьютера, которому необходимо произвести манипуляции с книгами.

* 1. **Состав функций**
     1. **Функция сохранения базы данных**

Эта функция сохраняет изменения в базе данных обновляя данные.

* + 1. **Функция поиска**

Эта функция ищет информацию о существующей книге.

* + 1. **Функция добавления**

Эта функция добавляет новую книгу, но не сохраняет данные.

* + 1. **Функция изменения**

Эта функция изменяет любую информацию уже существующей книги, но не сохраняет эту информацию.

1. **Условия выполнения программы**
   1. **Минимальный состав аппаратных средств**

ОС: Windows 10

Процессор: Как минимум 1 ГГц

ОЗУ: 256 МБ(для 32-разрядных систем).

Место на жестком диске: 103 МБ (для 32-разрядных систем) или 120 МБ (для 64-разрядных систем).

Дисплей: 911 x 112.

* 1. **Минимальный состав программных средств**

Дополнительные программные средства не требуются. Программа поставляется в exe.

* 1. **Требование к персоналу (пользователю)**

Конечный пользователь программы должен обладать практическими навыками работы с данной программой, а также иметь информацию о книгах.

1. **Выполнение программы**
   1. **Загрузка и запуск программы**

После загрузки на Компьютер необходимо создать папку где будет храниться программа, а также создать ярлык на рабочем столе.

Для запуска необходимо дважды кликнуть на «Library.exe» После чего подождать запуск.

Путь до каталога «Library.exe» должен содержать только латиницу.

Запустите программу «Library», откроется окно главного меню:



Рисунок 19. Главное меню

На рисунках 20, 21, 22, 23 и 24 представлены функции: изменения (Рисунок 20, 21), добавления (Рисунок 22, 23) и фильтрации (Рисунок 24).

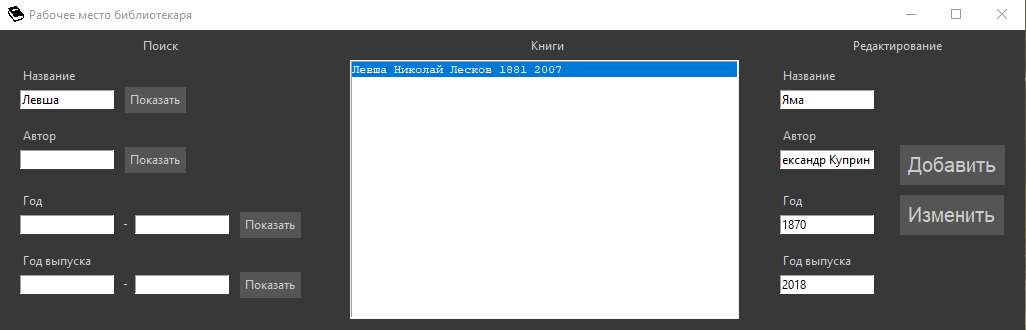


Рисунок 20. Заполнение полей

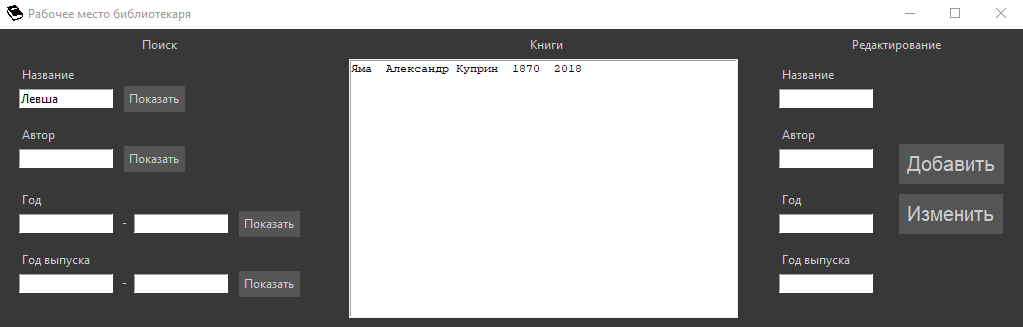


Рисунок 21. Применение новых значений

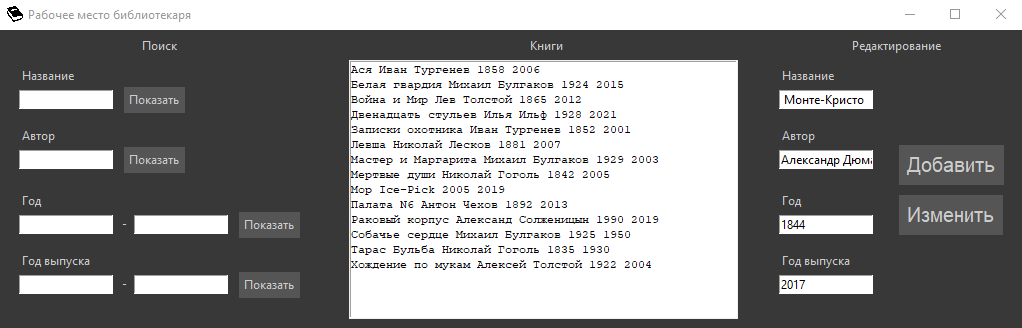


Рисунок 22. Заполнение полей

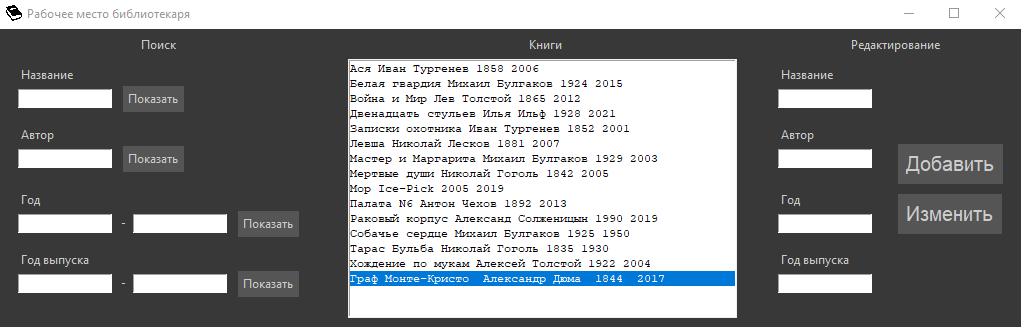
**

Рисунок 23. Результат добавления книги



Рисунок 24. Поиск книг

## **3.2 Элементы управления**

Интерфейсные кнопки:

Добавить – добавить новую книгу

Изменить – изменить существующую книгу

Показать – совершить поиск информации о книге

# **Заключение**

В ходе работы были проанализированы предметная область,

существующие разработки, посвященные данному направлению, получены

практические навыки по созданию UI с помощью библиотеки Tkinter.

Также планируется продолжать работу над данным проектом с целью

расширения возможностей и удобства приложения для пользователей. Планы

по доработкам представлены ниже.

To-do лист:

1. Расширить список данных для ввода, например имя издателя.

2. Адаптивность окна посредством динамического масштабирования и и

переноса элементов внутри программы.

3. Доработка интерфейса с целью обеспечения ввода без клавиатуры.

4. Создание кроссплатформенности.

5. Улучшения алгоритмов для ускорения процесса сбора данных.

6. Добавить иконки компаний.

# **Список литературы и интернет-источников**

1. Документация NoSQL. [Электронный ресурс]. Режим доступа: <https://docs.microsoft.com/ru-ru/azure/architecture/data-guide/big-data/non-relational-data>

(дата обращения: 06.03.2022)

1. Уроки по Tkinter. [Электронный ресурс]. Режим доступа: https://python-scripts.com/tkinter

(дата обращения: 08.03.2022)

1. Документация по Python. [Электронный ресурс]. Режим доступа:

https://pydocs.ru/

(дата обращения: 08.03.2022)

1. Ссылка на скачивание Python. [Электронный ресурс]. Режим доступа: https://www.python.org/downloads/

(дата обращения: 06.03.2022)

1. Сайт SQLite. [Электронный ресурс]. Режим доступа: https://www.sqlite.org/index.html

(дата обращения: 10.03.2022)

1. Ссылка на скачивание SQLite. [Электронный ресурс]. Режим доступа: https://www.sqlite.org/download.html

(дата обращения: 06.03.2022)

1. Сайт Visual Studio. [Электронный ресурс]. Режим доступа: <https://visualstudio.microsoft.com/ru/>

(дата обращения: 06.03.2022)

1. Документация по Visual Studio. [Электронный ресурс]. Режим доступа: <https://docs.microsoft.com/ru-ru/visualstudio/?view=vs-2022>

(дата обращения: 06.03.2022)

1. Ссылка на скачивание Visual Studio. [Электронный ресурс]. Режим доступа: <https://visualstudio.microsoft.com/ru/downloads/>

(дата обращения: 06.03.2022)

1. Подключение к базе данных. [Электронный ресурс]. Режим доступа: https://pythonru.com/biblioteki/vvedenie-v-sqlite-python

(дата обращения: 06.03.2022)

# **Приложения**

**Листинг 6. LibTK.py**

from tkinter import \*

from tkinter import ttk

import sqlite3

#----------------------------------------------------------

# Функци для кнопок "Показать"

def main():

def getk():

conn = sqlite3.connect('LibTable')

curs = conn.cursor()

\*gridAuto, = curs.execute("""SELECT \*

FROM Lib

WHERE Year BETWEEN ? AND ?

ORDER BY Year DESC""",\

(Yearbox1.get(), Yearbox2.get()))

mainlist.delete(0, END)

for i in gridAuto:

\*i, = i

rowtext.set(str(i[0]) + ' ' + str(i[1]) + ' ' + \

str(i[2]) + ' ' + str(i[3]))

mainlist.insert(END, rowtext.get())

def geth():

conn = sqlite3.connect('LibTable')

curs = conn.cursor()

\*gridAuto, = curs.execute("""SELECT \*

FROM Lib

WHERE Price BETWEEN ? AND ?

ORDER BY Price DESC""",\

(Pricebox1.get(), Pricebox2.get()))

mainlist.delete(0, END)

for i in gridAuto:

\*i, = i

rowtext.set(str(i[0]) + ' ' + str(i[1]) + ' ' + \

str(i[2]) + ' ' + str(i[3]))

mainlist.insert(END, rowtext.get())

def getplayer():

conn = sqlite3.connect('LibTable')

curs = conn.cursor()

\*gridAuto, = curs.execute("""SELECT \*

FROM Lib

WHERE Name LIKE ?

ORDER BY Name""",\

('%' + namebox.get() + '%',))

mainlist.delete(0, END)

for i in gridAuto:

\*i, = i

rowtext.set(str(i[0]) + ' ' + str(i[1]) + ' ' + \

str(i[2]) + ' ' + str(i[3]))

mainlist.insert(END, rowtext.get())

print(gridAuto)

def getAuthor():

conn = sqlite3.connect('LibTable')

curs = conn.cursor()

\*gridAuto, = curs.execute("""SELECT \*

FROM Lib

WHERE HP LIKE ?

ORDER BY HP""",\

('%' + HPbox1.get() + '%',))

mainlist.delete(0, END)

for i in gridAuto:

\*i, = i

rowtext.set(str(i[0]) + ' ' + str(i[1]) + ' ' + \

str(i[2]) + ' ' + str(i[3]))

mainlist.insert(END, rowtext.get())

def savedBD():

conn = sqlite3.connect('LibTable')

curs = conn.cursor()

curs.execute("DELETE FROM Lib")

a = []

for l, i in enumerate(mainlist.get(0, END)):

a.append([])

for j in i.split(' '):

a[l].append(j)

print(a)

curs.executemany("INSERT INTO Lib VALUES (?, ?, ?, ?)", a)

conn.commit()

def update():

conn = sqlite3.connect('LibTable')

curs = conn.cursor()

\*grid, = curs.execute("""SELECT \*

FROM Lib""")

mainlist.delete(0, END)

for i in grid:

mainlist.insert(END, str(i[0]) + ' ' + str(i[1]) + ' ' + \

str(i[2]) + ' ' + str(i[3]))

def add():

mainlist.insert(END, NameVar1.get() + ' ' \

+ AuthorVar2.get() + ' ' \

+ YearVar3.get() + ' ' \

+ PriceVar4.get())

NameVar1.set('')

AuthorVar2.set('')

YearVar3.set('')

PriceVar4.set('')

savedBD()

def remove(self):

select = list(mainlist.curselection())

mainlist.delete(select)

savedBD()

def change():

select = list(mainlist.curselection())

mainlist.delete(select)

mainlist.insert(select, NameVar1.get() + ' ' \

+ AuthorVar2.get() + ' ' \

+ YearVar3.get() + ' ' \

+ PriceVar4.get())

NameVar1.set('')

AuthorVar2.set('')

YearVar3.set('')

PriceVar4.set('')

savedBD()

#----------------------------------------------------------

# Создание окна, иконки и переменных

tk = Tk()

tk.title('Рабочее место библиотекаря')

tk.geometry('1025x300+700+300')

tk["bg"] = "gray22"

tk.call('wm', 'iconphoto', tk.\_w, PhotoImage(file='lib-logo.png'))

#logo = Label(tk, text='LIBRARY', font='Arial 45', background="gray22", foreground="#ccc").place(x=10, y=50)

rowtext = StringVar()

nameplayer = StringVar()

NameVar1 = StringVar()

AuthorVar2 = StringVar()

YearVar3 = StringVar()

PriceVar4 = StringVar()

grid = ['', '', '', '']

#----------------------------------------------------------

# Создание названий разделов и кнопок

nametext = Label(tk, text='Название', background="gray22", foreground="#ccc").place(x=20, y=35)

HPtext = Label(tk, text='Автор', background="gray22", foreground="#ccc").place(x=20, y=95)

Yeartext = Label(tk, text='Год', background="gray22", foreground="#ccc").place(x=20, y=160)

Pricetext = Label(tk, text='Год выпуска', background="gray22", foreground="#ccc").place(x=20, y=220)

searchtext = Label(tk, text='Поиск', background="gray22", foreground="#ccc").place(x=140, y=5)

searchtext = Label(tk, text='Редактирование', background="gray22", foreground="#ccc").place(x=850, y=5)

#----------------------------------------------------------

# Создание окн ввода и кнопок "Показать"

namebox = Entry(tk, width=15, textvariable=nameplayer)

namebox.place(x=20, y=60)

HPbox1 = Entry(tk, width=15)

HPbox1.place(x=20, y=120)

Yearbox1 = Entry(tk, width=15)

Yearbox1.place(x=20, y=185)

Yeartext = Label(tk, text='-', background="gray22", foreground="#ccc").place(x=120, y=183)

Yearbox2 = Entry(tk, width=15)

Yearbox2.place(x=135, y=185)

Pricebox1 = Entry(tk, width=15)

Pricebox1.place(x=20, y=245)

Pricetext = Label(tk, text='-', background="gray22", foreground="#ccc").place(x=120, y=243)

Pricebox2 = Entry(tk, width=15)

Pricebox2.place(x=135, y=245)

namebutton = Button(tk, text='Показать', relief = 'flat', background="#555", foreground="#ccc", activebackground="#333333", command=getplayer).place(x=125, y=57)

HPbutton = Button(tk, text='Показать', relief = 'flat', background="#555", foreground="#ccc", activebackground="#333333", command=getAuthor).place(x=125, y=117)

Yearbutton = Button(tk, text='Показать', relief = 'flat', background="#555", foreground="#ccc", activebackground="#333333", command=getk).place(x=240, y=182)

Pricebutton = Button(tk, text='Показать', relief = 'flat', background="#555", foreground="#ccc", activebackground="#333333", command=geth).place(x=240, y=242)

Namebox3 = Entry(tk, width=15, textvariable=NameVar1).place(x=780, y=60)

Authorbox3 = Entry(tk, width=15, textvariable=AuthorVar2).place(x=780, y=120)

Yearbox3 = Entry(tk, width=15, textvariable=YearVar3).place(x=780, y=185)

Pricebox3 = Entry(tk, width=15, textvariable=PriceVar4).place(x=780, y=245)

nametext = Label(tk, text='Название', background="gray22", foreground="#ccc").place(x=780, y=35)

HPtext = Label(tk, text='Автор', background="gray22", foreground="#ccc").place(x=780, y=95)

Yeartext = Label(tk, text='Год', background="gray22", foreground="#ccc").place(x=780, y=160)

Pricetext = Label(tk, text='Год выпуска', background="gray22", foreground="#ccc").place(x=780, y=220)

addbutton = Button(tk, text='Добавить', relief = 'flat', font='Arial 15', background="#555", foreground="#ccc", activebackground="#333333", command=add).place(x=900, y=115)

delbutton = Button(tk, text='Изменить', relief = 'flat', font='Arial 15', background="#555", foreground="#ccc", activebackground="#333333", command=change).place(x=900, y=165)

#----------------------------------------------------------

# Создание окна для показа таблици

maintext = Label(tk, text = 'Книги', background="gray22", foreground="#ccc").place(x=528, y=5)

mainlist = Listbox(tk, width=55, height=17, font='Courier 8')

mainlist.place(x=350, y=30)

#mainlist.bind('<Delete>', remove)

update()

mainloop()

if \_\_name\_\_ == '\_\_main\_\_':

main()

**Листинг 7. LibData.py**

import sqlite3

# Создание списка таблици

''' Название Автор Год Год выпуска '''

text = [('Мастер и Маргарита', 'Михаил Булгаков','1929','2003'),

('Ася', 'Иван Тургенев','1858','2006'),

('Мертвые души', 'Николай Гоголь','1842 ','2005'),

('Хождение по мукам', 'Алексей Толстой','1922','2004'),

('Записки охотника', 'Иван Тургенев','1852','2001'),

('Война и Мир', 'Лев Толстой','1865 ','2012'),

('Левша', 'Николай Лесков','1881','2007'),

('Двенадцать стульев', 'Илья Ильф','1928','2021'),

('Белая гвардия', 'Михаил Булгаков','1924','2015'),

('Палата N6', 'Антон Чехов','1892','2013'),

('Собачье сердце', 'Михаил Булгаков','1925','1950'),

('Мор', 'Ice-Pick','2005','2019'),

('Тарас Бульба', 'Николай Гоголь','1835','1930')]

# Создание и запись списка в таблицу

conn = sqlite3.connect('LibTable')

curs = conn.cursor()

curs.execute("""CREATE TABLE Lib(

Name TEXT,

HP TEXT,

Year INTEGER,

Price INTEGER)""")

curs.executemany("INSERT INTO Lib VALUES (?, ?, ?, ?)", text)

conn.commit()

1. 1) ГОСТ 19.101-77 ЕСПД. Виды программ и программных документов [↑](#footnote-ref-1)
2. 2) ГОСТ 19.103-77 ЕСПД. Обозначение программ и программных документов [↑](#footnote-ref-2)
3. 3) ГОСТ 19.104-78\* ЕСПД. Основные надписи [↑](#footnote-ref-3)
4. 4) ГОСТ 19.105-78\* ЕСПД. Общие требования к программным документам [↑](#footnote-ref-4)
5. 5) ГОСТ 19.106-78\* ЕСПД. Общие требования к программным документам, выполненным печатным способом [↑](#footnote-ref-5)
6. 6) ГОСТ 19.505-79\* ЕСПД. Руководство оператора. Требования к содержанию и оформлению [↑](#footnote-ref-6)
7. 7) ГОСТ 19.604-78\* ЕСПД. Правила внесения изменений в программные документы, выполненные печатным способом [↑](#footnote-ref-7)