



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение  
высшего образования

**«МИРЭА – Российский технологический университет»  
РТУ МИРЭА**

Институт Информационных Технологий

Кафедра Корпоративных Информационных Систем

**КУРСОВОЙ ПРОЕКТ (РАБОТА)**

по дисциплине разработка программных приложений  
*(наименование дисциплины)*

**Тема курсового проекта (работы)** разработка информационной системы «Библиотека»

Студент группы ИКБО-08-18 Валяев Данила Андреевич  
*(учебная группа, фамилия, имя, отчество студента)* *(подпись студента)*

**Руководитель курсового проекта (работы)** ст. преп. Мирзоян Д.И.  
*(подпись руководителя)*

Работа представлена к защите « \_\_\_\_ » \_\_\_\_\_ 20 \_\_\_\_ г.

Допущен к защите « \_\_\_\_ » \_\_\_\_\_ 20 \_\_\_\_ г.



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение  
высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт Информационных Технологий

Кафедра Корпоративных Информационных Систем

Утверждаю

Заведующий кафедрой КИС

\_\_\_\_\_ Андрианова Е.Г.

« \_\_\_\_ » \_\_\_\_\_ 20 \_\_\_\_ г.

**ЗАДАНИЕ**

на выполнение курсового проекта (работы) по дисциплине

«Разработка программных приложений»

Студент Валяев Данила Андреевич Группа ИКБО-08-18

Тема разработка информационной системы «Библиотека»

Исходные данные: описание языка и среды разработки на языке Python, стандарты оформления программного кода, регламент работы библиотеки, шаблоны описания библиотечного фонда.

Перечень вопросов, подлежащих разработке, и обязательного графического материала:

1. ER модель предметной области
2. Состав, атрибуты классов и их типы
3. Формат долговременного хранения данных в БД
4. Пользовательский интерфейс приложения
5. Набор автоматизируемых функций

Срок представления к защите курсового проекта (работы): до « \_\_\_\_ » \_\_\_\_\_ 201 \_\_\_\_ г.

Задание на выполнение курсовой проект (работу) выдал \_\_\_\_\_ ( Мирзоян Д.И. )  
Подпись руководителя Ф.И.О. руководителя

Задание на курсовой проект (работу) получил \_\_\_\_\_ ( Валяев Д.А. )  
Подпись обучающегося Ф.И.О. исполнителя

« \_\_\_\_ » \_\_\_\_\_ 20 \_\_\_\_ г.

## Техническое задание:

### *1. Наименование работы*

Разработка информационной системы «Библиотека».

### *2. Исполнитель*

Валяев Данила Андреевич

Шифр: 18И1192

### *Основание для разработки:*

Учебный план бакалавриата по направлению 09.03.04 «Программная инженерия».

### *3. Цель и назначение работы*

Данный продукт разработан с целью помочь работникам сферы регулирования библиотечного фонда для удобного хранения и отображения информации о книгах, находящихся в библиотеке.

### *4. Технические требования*

Рассмотреть процессы выдачи и возврата материалов, пополнения фондов.

Автоматизировать отчеты о находящихся на руках материалах, о частоте выдачи материала, среднем времени пребывания материала на руках.

### *5. Содержание работы*

В результате выполнения работы решаются следующие задачи:

1. Составление теоретического материала
2. Разработка структуры системы
3. Оформление Расчетно-пояснительной записки
4. Программная реализация системы
5. Защита работы и демонстрация работы системы

### *6. Порядок сдачи работы*

Выполнения работы (РПЗ и ИС в соответствии со стандартами кафедры КИС)

**Руководитель**

**Исполнитель**

(подпись, дата)

(подпись, дата)

## **Аннотация**

В данной курсовой работе рассмотрено создание информационной системы «Библиотека», приложение типа Desktop application для целевой операционной системы Windows. Рассмотрены процессы: анализа предметной области, проектирования приложения, реализации приложения и тестирования приложения, написания инструкции для конечного пользователя.

Была написана инструкция для пользователя, в которой подробно и пошагово расписаны действия пользователя для осуществления типовых сценариев работы с системой «Библиотека».

## Содержание

Введение .....	8
1. Анализ предметной области .....	10
1.1. Исследование предметной области .....	10
1.2. Обзор существующих в предметной области технологий.....	11
1.3. Вывод к первой главе.....	14
2. Проектирование приложения .....	14
2.1. Разработка пользовательского интерфейса.....	14
2.2. Описание структуры программы .....	14
2.3. Алгоритмы, используемые в программе.....	15
2.4. Проектирование системы хранения .....	15
2.5. Проектирование программного приложения.....	16
3. Реализация приложения.....	17
3.1. Реализация алгоритмов.....	17
3.2. Реализация программного приложения .....	21
3.3. Вывод к третьей главе.....	24
4. Тестирование приложения.....	24
4.1. Тестирование программного приложения .....	24
5. Инструкция .....	28
5.1. Инструкция для пользователя.....	28
6. Заключение .....	33
7. Список используемых источников .....	34
8. Приложение.....	35

# Введение

В данной курсовой работе представлена реализация информационной системы «Библиотека». В нее входят система хранения книг, система выдачи и возврата книг, система сортировки по разным признакам, системы пополнения и списания библиотечного фонда.

Программная часть реализована на языке Python, с использованием библиотеки Tkinter для работы с графическим интерфейсом в среде разработки PyCharm и библиотеки sqlite3 для работы с базой данных.

Ведение учета книг в библиотеки – сложный процесс и для его упрощения, отличным решением является разработка специального приложения. Работник библиотеки выдать или принять книгу, добавить информацию о недавно поступивших книгах, отсортировать книги по различным признакам, таким как:

- идентификационный номер;
- автор;
- название;
- год издания;
- количество экземпляров в библиотеке;

Так же программное приложение предоставляет возможность автоматической генерации отчетов:

- о частоте выдачи материала;
- о находящихся на руках в данный момент материалах;
- о среднем времени нахождения материала на руках;

Программное приложение имеет 3 типа пользователей: администратор, работник и пользователь.

Возможности пользователя:

- сортировка списка книг;
- выдача и возврат материала в библиотеку;

Возможности работника:

- запрашивать отчеты;
- выдавать и принимать материалы;
- сортировать список материалов;
- добавлять и удалять информацию о материале;

Возможности администратора:

- запрашивать отчеты;
- выдавать и принимать материалы;
- сортировать список материалов;
- добавлять и удалять информацию о материале;
- пополнять библиотечный фонд;

# **1. Анализ предметной области**

## **1.1. Исследование предметной области**

Программное приложение мониторинга библиотечного фонда разработано как настольное приложение для компьютеров, с графическим пользовательским интерфейсом. Информация о книгах хранится в базе данных.

Языком реализации проекта является Python. Также при разработке использовалась стандартная библиотека Tkinter для работы с графическим интерфейсом, и библиотека sqlite3 для работы с базой данных.

Целевой операционной системой для использования программы является Windows.

### **1. Находящиеся в библиотеке материалы**

Все книги, находящиеся в библиотеке, имеют определенные характеристики

К этим характеристикам относятся:

- Id книги;
- Название;
- Автор книги;
- Год издания;
- Кол-во книг в библиотеке;

### **2. Выданные материалы**

Все книги, находящиеся на выдаче, имеют определенные характеристики

К этим характеристикам относятся:

- Id книги;
- Название;
- Автор книги;



- Год издания;
- Кол-во книг в библиотеке;

### **3. Функционал программного приложения**

Также при анализе предметной области был выявлен функционал программного приложения.

К нему относятся:

- Добавление информации о книге;
- Удаление информации о книге;
- Вывод информации о книге;
- Сортировка книг по различным критериям;
- Выдача и возврат книг в библиотеку;
- Возможность регистрации;
- Формирование отчета о выданных книгах;
- Формирование отчета о частоте выдачи книг;
- Формирование отчета о среднем времени пребывания на руках;
- Возможность авторизации как администратор, пользователь и работник библиотеки;

## **1.2. Обзор существующих в предметной области технологий**

### **1. Tkinter**

Tkinter – это пакет для Python, предназначенный для работы с библиотекой Tk. Библиотека Tk содержит компоненты графического интерфейса пользователя (graphical user interface – GUI), написанные на языке программирования Tcl.

В настоящее время почти все приложения, которые создаются для конечного пользователя, имеют GUI. Редкие программы, подразумевающие взаимодействие с человеком, остаются консольными. В предыдущих двух курсах мы писали только консольные программы.

Существует множество библиотек GUI. Tk далеко не самая популярная, хотя с ее использованием написано не мало проектов. Однако по ряду причин она была выбрана для Python по-умолчанию. Установочный файл Питона обычно уже включает пакет tkinter в составе стандартной библиотеки наряду с другими модулями.

Не вдаваясь в подробности, Tkinter можно охарактеризовать как переводчик с языка Python на язык Tcl. Вы пишете программу на Python, а код модуля tkinter у вас за спиной переводит ваши инструкции на язык Tcl, который понимает библиотека Tk.

Библиотека Tkinter была выбрана мной из-за того, что в отличие от остальных библиотек для работы с графическим интерфейсом, она не содержит в себе ничего лишнего, благодаря чему проста в использовании.

## 2. SQL

Для хранения информации о всех устройствах, поступивших в сервисный центр будет использован SQL. SQL —декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных. При всех своих изменениях SQL остаётся самым распространённым лингвистическим средством для взаимодействия прикладного программного обеспечения с базами данных. В то же время современные СУБД, а также информационные системы, использующие СУБД, предоставляют пользователю развитые средства визуального построения запросов. Несмотря на наличие диалектов и различий в синтаксисе, в большинстве своём тексты SQL-запросов, содержащие DDL и DML, могут быть достаточно легко перенесены из одной СУБД в другую. Существуют системы, разработчики которых изначально ориентировались на применение по меньшей мере нескольких СУБД.

Естественно, что при применении некоторых специфичных для реализации возможностей такой переносимости добиться уже очень трудно. С помощью SQL программист описывает только то, какие данные нужно извлечь или модифицировать. То, каким образом это сделать, решает СУБД непосредственно при обработке SQL-запроса. Однако не стоит думать, что это полностью универсальный принцип — программист описывает набор данных для выборки или модификации, однако ему при этом полезно представлять, как СУБД будет разбирать текст его запроса. Чем сложнее сконструирован запрос, тем больше он допускает вариантов написания, различных по скорости выполнения, но одинаковых по итоговому набору данных.

### **3. SQLite**

Слово «встраиваемый» (embedded) означает, что SQLite не использует парадигму клиент-сервер, то есть движок SQLite не является отдельно работающим процессом, с которым взаимодействует программа, а представляет собой библиотеку, с которой программа компонуется, и движок становится составной частью программы. Таким образом, в качестве протокола обмена используются вызовы функций библиотеки SQLite. Такой подход уменьшает накладные расходы, время отклика и упрощает программу. SQLite хранит всю базу данных (включая определения, таблицы, индексы и данные) в единственном стандартном файле на том компьютере, на котором исполняется программа. Простота реализации достигается за счёт того, что перед началом исполнения транзакции записи весь файл, хранящий базу данных, блокируется; ACID-функции достигаются в том числе за счёт создания файла журнала.

Несколько процессов или потоков могут одновременно без каких-либо проблем читать данные из одной базы. Запись в базу можно осуществить только в том случае, если никаких других запросов в данный момент не

обслуживается; в противном случае попытка записи оканчивается неудачей, и в программу возвращается код ошибки. Другим вариантом развития событий является автоматическое повторение попыток записи в течение заданного интервала времени.

### 1.3. Вывод к первой главе

В результате анализа предметной области, структур данных и алгоритма сортировки была построена структура всего проекта.

## 2. Проектирование приложения

### 2.1. Разработка пользовательского интерфейса

Всего приложение содержит одно окно. Графический интерфейс проектировался с целью удобства и понятности использования. Пользовательский интерфейс состоит из пяти полей ввода для заполнения информации о книге, поля ввода id для удаления, пятнадцати основных кнопок, реализующих функции добавления, удаления, заполнения, сортировки по различным критериям, выдачи, возврата книг.

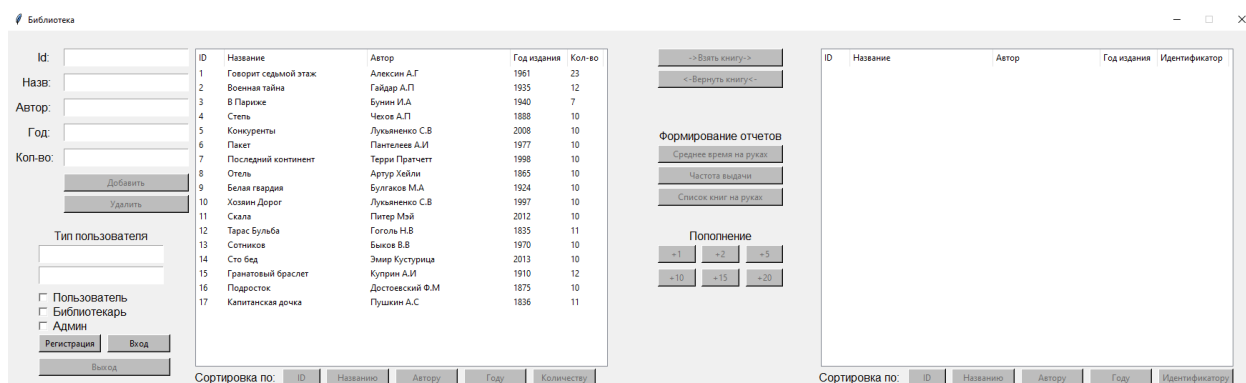


Рис. 1 – Пользовательский интерфейс приложения

### 2.2. Описание структуры программы

Логическая структура программы подразумевает использование одной из парадигм программирования – процедурного программирования.

Процедурный стиль программирования предназначен для различного рода дополнительных функций или операций, где производятся лишь видоизменения и преобразования над уже созданными объектами классов и

переменными. ПП позволяет сохранить простоту читаемости кода, когда необходимо создать функции, внесение которых в конкретный класс неуместно, а создание нового усложняет читаемость.

### **2.3. Алгоритмы, используемые в программе**

Основными алгоритмами данного программного приложения являются операции добавления, удаления, выдачи и возврата книг, возможность авторизации как администратор, пользователь и работник библиотеки. Также к основным алгоритмам относятся операции сортировки книг по различным критериям, таким как:

- Идентификатор;
- Название;
- Автор;
- год издания;
- количество экземпляров в библиотеке;

### **2.4. Проектирование системы хранения**

Во многих проектах возникает потребность в хранении немалого объема данных, но в то же время использование СУБД является слишком накладным из-за сложности развертывания приложения. И тут на помощь приходит такая прекрасная вещь как SQLite – компактная встраиваемая база данных с которой работает и наше приложение.

Данные о книгах в библиотеке хранятся в базе данных с расширением .db, при запуске программы, она автоматически заполняет таблицу, предназначенную для хранения списка книг, элементами из базы данных

ID	Название	Автор	Год издания	Кол-во
1	Говорит седьмой этаж	Алексин А.Г	1961	23
2	Военная тайна	Гайдар А.П	1935	12
3	В Париже	Бунин И.А	1940	7
4	Степь	Чехов А.П	1888	10
5	Конкуренты	Лукьяненко С.В	2008	10
6	Пакет	Пантелеев А.И	1977	10
7	Последний континент	Терри Пратчетт	1998	10
8	Отель	Артур Хейли	1865	10
9	Белая гвардия	Булгаков М.А	1924	10
10	Хозяин Дорог	Лукьяненко С.В	1997	10
11	Скала	Питер Мэй	2012	10
12	Тарас Бульба	Гоголь Н.В	1835	11
13	Сотников	Быков В.В	1970	10
14	Сто бед	Эмир Кустурица	2013	10
15	Гранатовый браслет	Куприн А.И	1910	12
16	Подросток	Достоевский Ф.М	1875	10
17	Капитанская дочка	Пушкин А.С	1836	11

Рис. 2 – Вывод информации из базы данных на пользовательский интерфейс.

## 2.5. Проектирование программного приложения

Во время разработки следует разбить программное обеспечение на несколько областей по смыслу, для удобства пользования.

Пользовательский ввод должно проверяться на основе допустимых значений и выводить ошибку если что-то пошло не так.

Для того чтобы точнее проинформировать пользователя о некорректном вводе, следует разбить одну большую проверку, на несколько меньших, что позволит увеличить точность отображаемых сообщений с предупреждением об ошибке ввода.

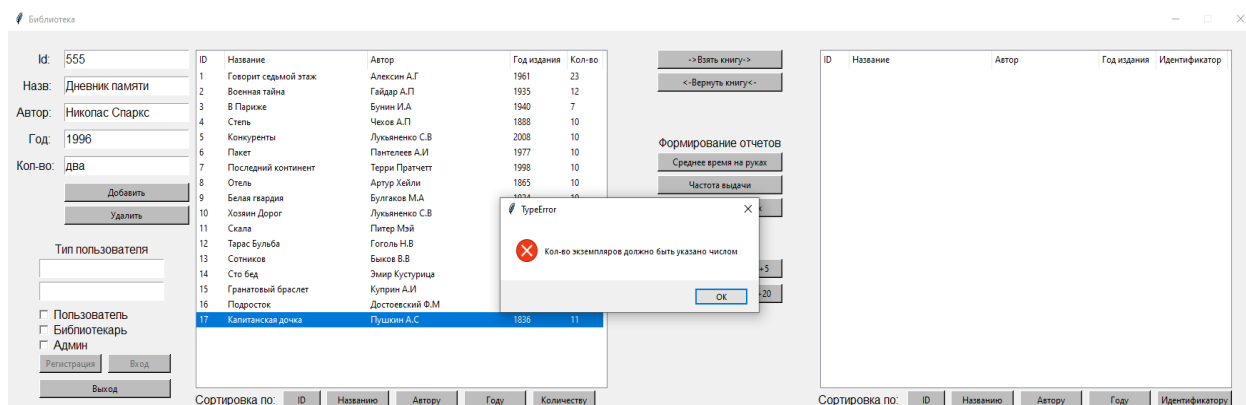


Рис. 3 – Окно ошибки при некорректном вводе кол-ва экземпляров

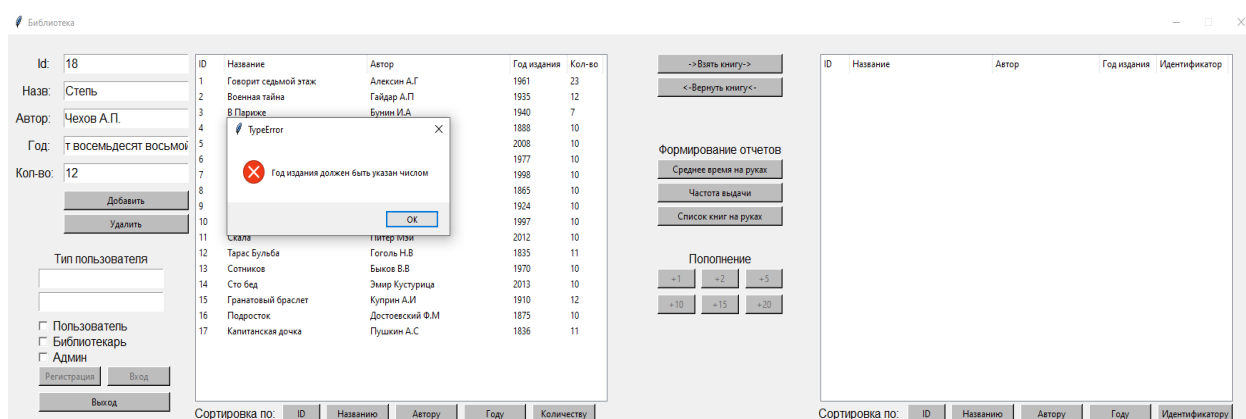


Рис. 4 – Окно ошибки при некорректном вводе года издания

## 3. Реализация приложения

### 3.1. Реализация алгоритмов

#### Заполнение таблицы, содержащей список книг

Сначала таблица, хранящая список книг очищается, далее производится ее заполнение из базы данных.

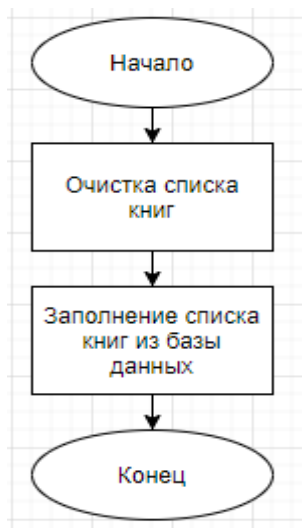


Рис. 5 – Заполнение таблицы, содержащей список книг

### Добавление элемента

Функция добавления, получает на вход информацию о книге. Далее программа проверяет правильность введенных данных и уникальность идентификатора книги, если данные введены корректно и идентификатор уникален, программа добавляет информацию о книге в базу данных.

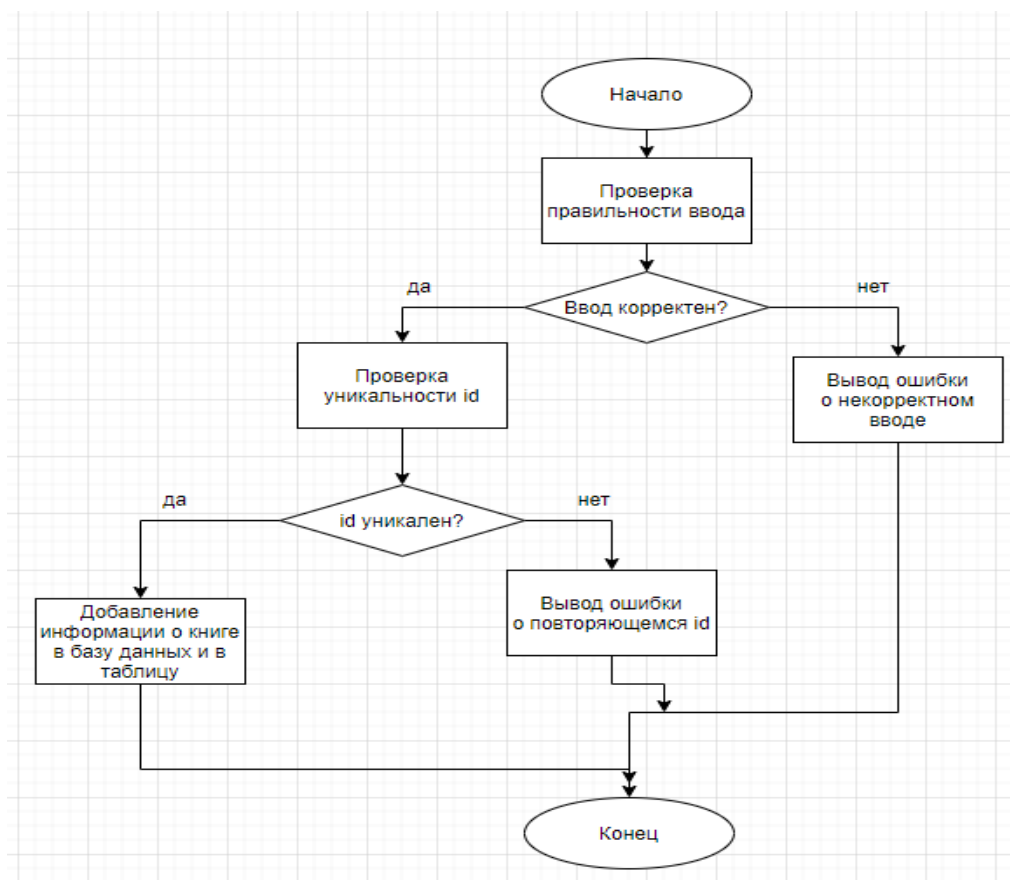


Рис. 6 – Графическое изображение добавления элемента



## Удаление элемента

Функция удаления, получает на вход выбранный элемент из таблицы, который необходимо удалить. Следующим шагом она проверяет выбран ли какой-либо элемент, если элемент выбран, программа находит в базе данных запись с идентификатором как у выбранного элемента и производит удаление, далее программа удаляет запись из таблицы.

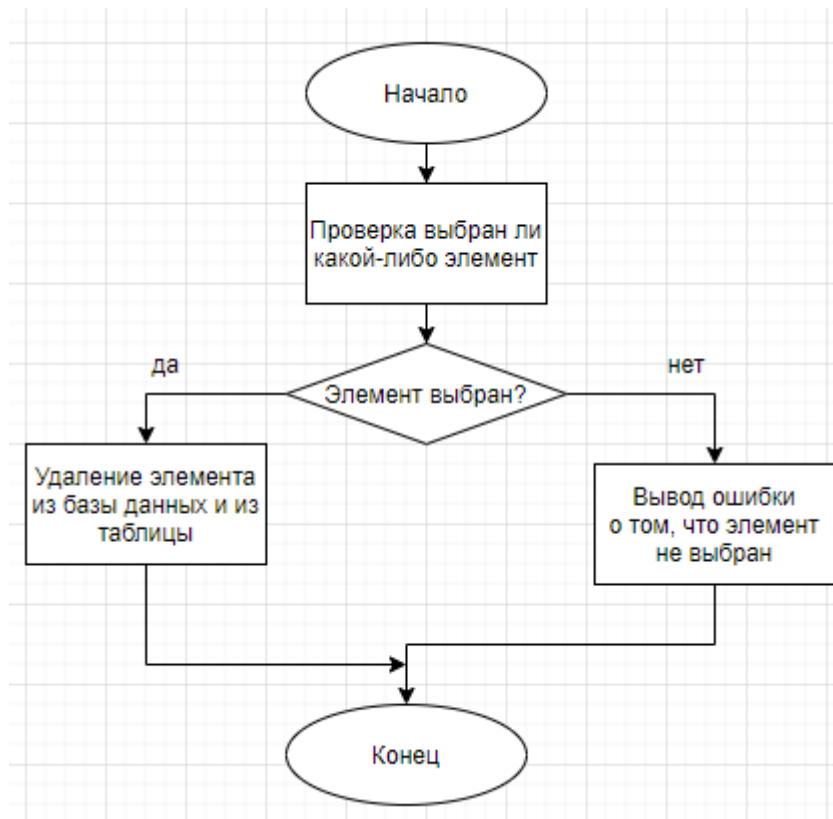


Рис. 7 – Графическое изображение удаления элемента

## Регистрация пользователя

Функция регистрации пользователя получает на вход логин, пароль и тип пользователя, далее программа проверяет зарегистрирован ли пользователь с введенным логином, если пользователь отсутствует, программа заносит информацию о новом пользователе в базу данных.

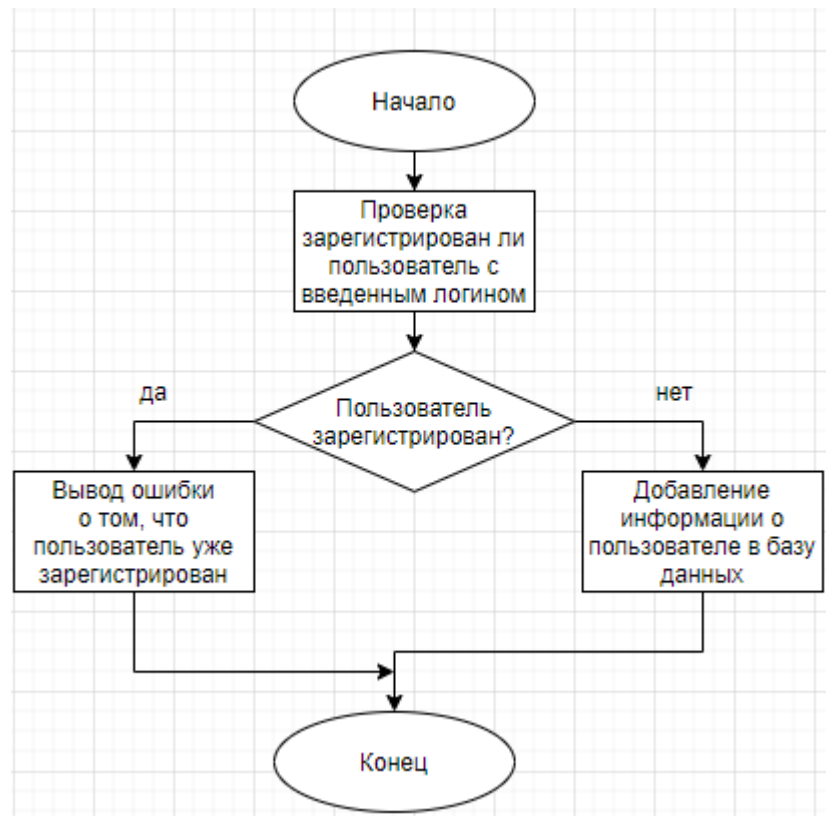


Рис. 8 – Графическое изображение регистрации пользователя

### **Авторизация пользователя**

Функция авторизации пользователя получает на вход логин и пароль, далее программа проверяет зарегистрирован ли пользователь с введенным логином, если пользователь существует, программа проверяет правильность ввода пароля, если пароль введен верно, программа разрешает доступ к библиотеке.

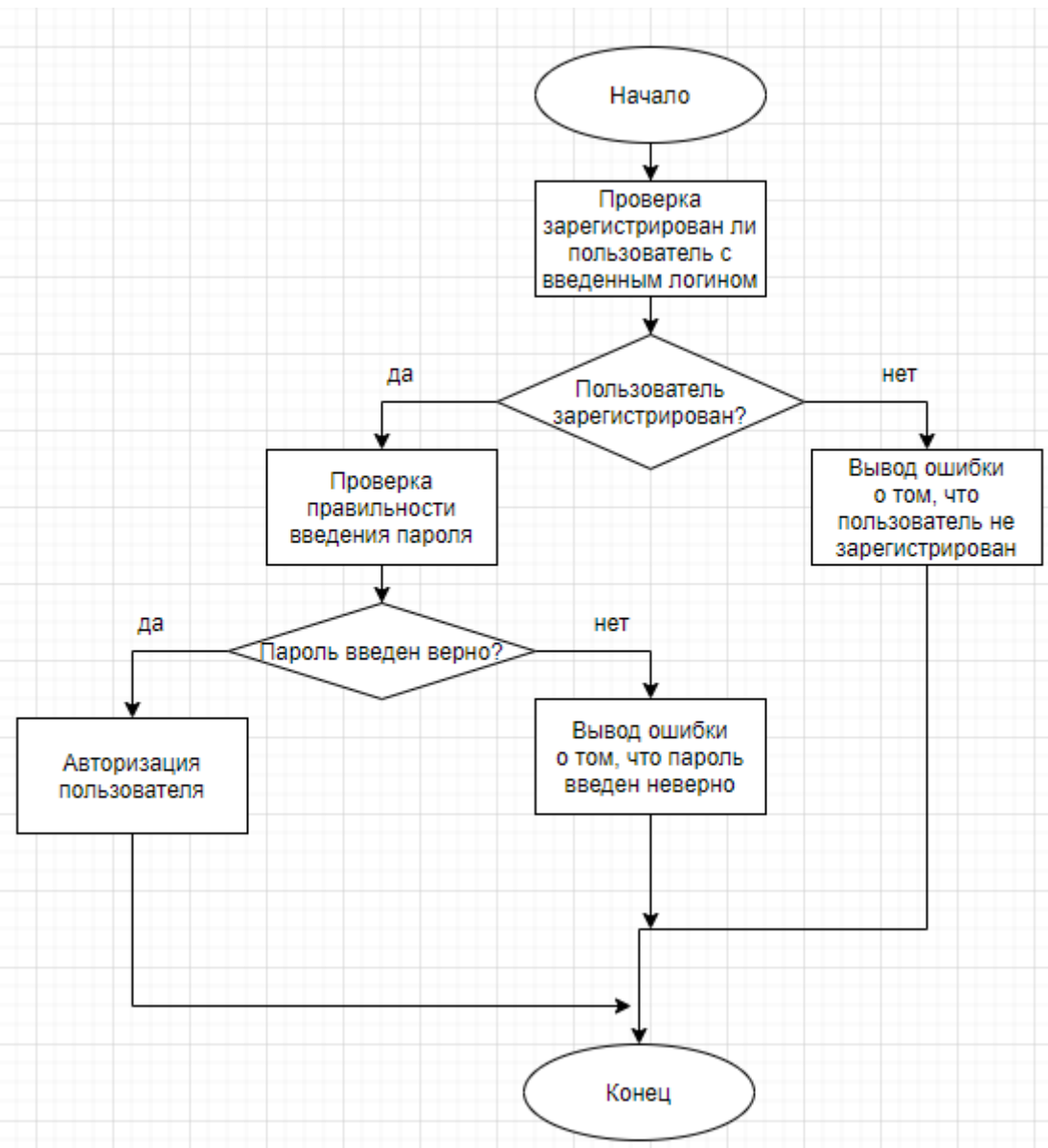


Рис. 9 – Графическое изображение авторизации пользователя

### 3.2. Реализация программного приложения

У каждого поля свой тип данных, и что бы программа не прекращала работу при некорректном вводе, будет обработана и выведена ошибка, что введенные данные некорректны, и вы сможете повторить ввод.

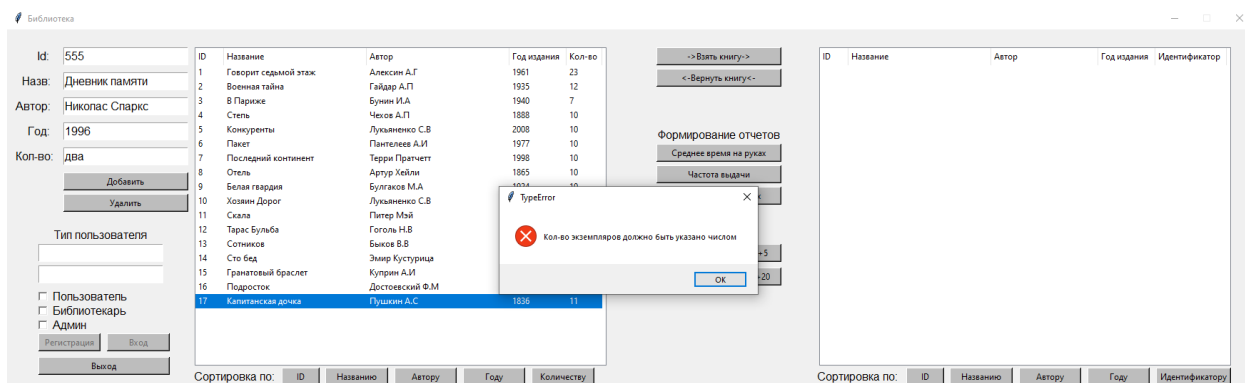


Рис. 10 – Окно ошибки при некорректном вводе кол-ва экземпляров

При добавлении информации о новой книге необходимо заполнить все поля, в случае если этого не сделать будет выведена соответствующая ошибка.

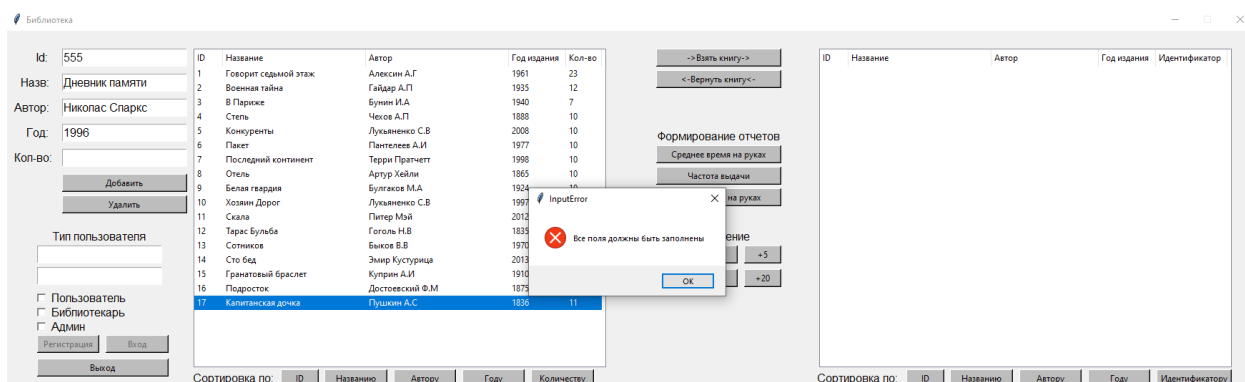


Рис. 11 – Окно ошибки при неполном заполнении

Если все поля заполнены корректно, информация о книге будет добавлена в список и выведена в окно приложения.

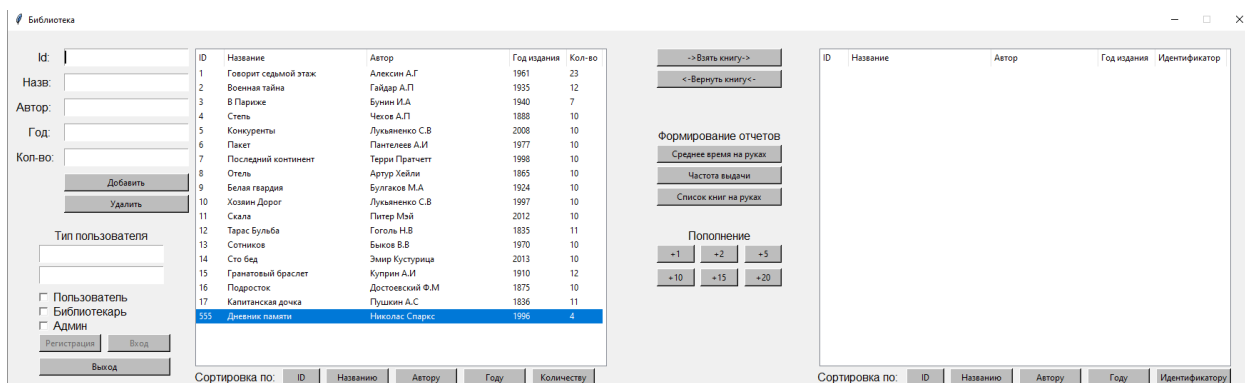


Рис. 12 – Корректное добавление информации о книге

При попытке добавления в список книги с id, которое уже находится в списке, будет выведена ошибка.

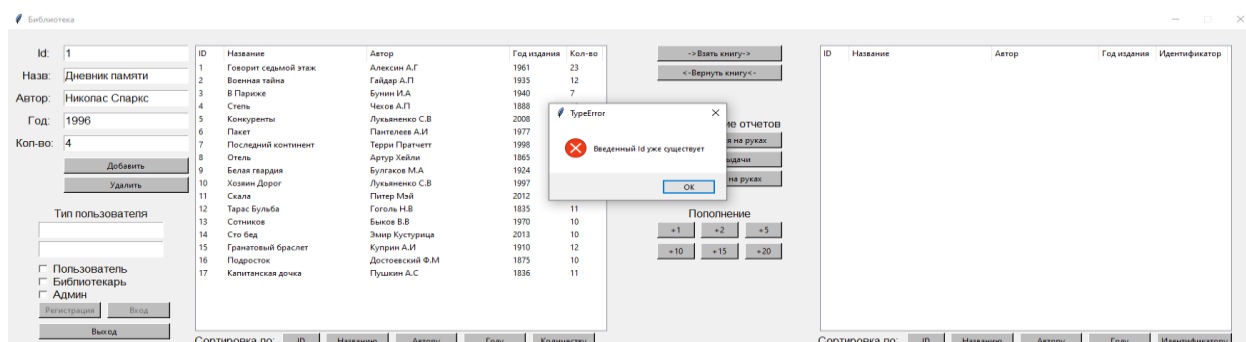


Рис. 13 – Окно ошибки при добавлении повторного id

При попытке удаления книги, не выбрав ее, будет выведена ошибка.

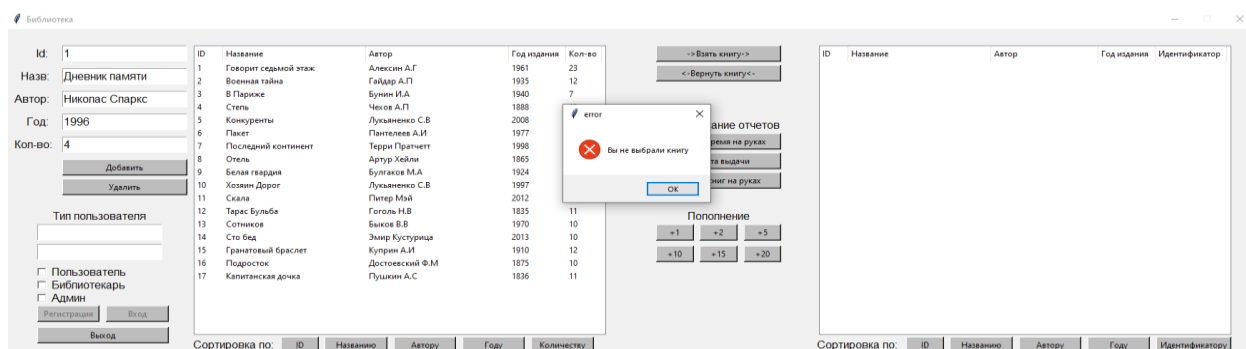


Рис. 14 – Окно ошибки при отсутствии выбора элемента для удаления

В случае если пользователю необходимо просмотреть список книг, находящихся на руках, необходимо нажать на кнопку «Список книг на руках» и в правой части окна будет отображен результат.

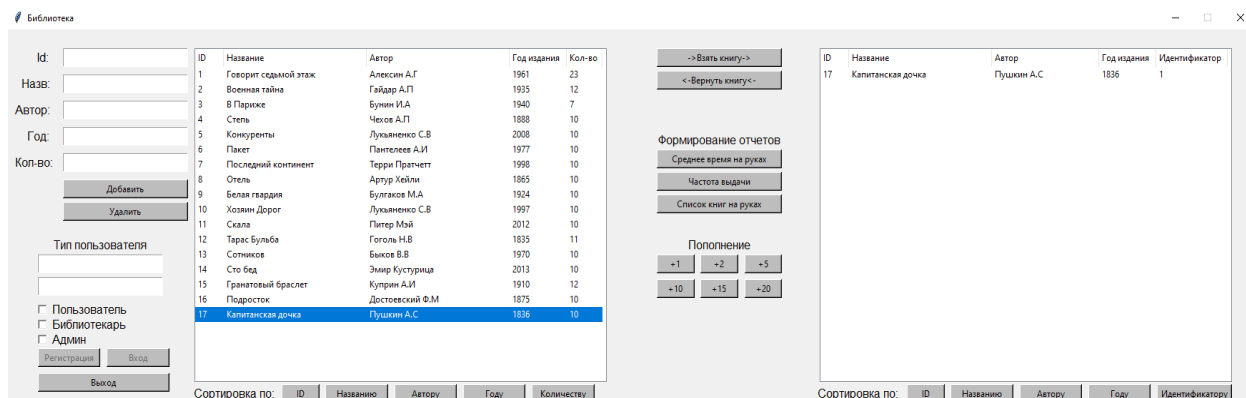


Рис. 15 – Просмотр списка выданных книг

Для более удобного просмотра содержимого библиотеки, программное приложение содержит функцию сортировки, для ее использования необходимо нажать на одну из кнопок, находящихся в нижней части окна программы.

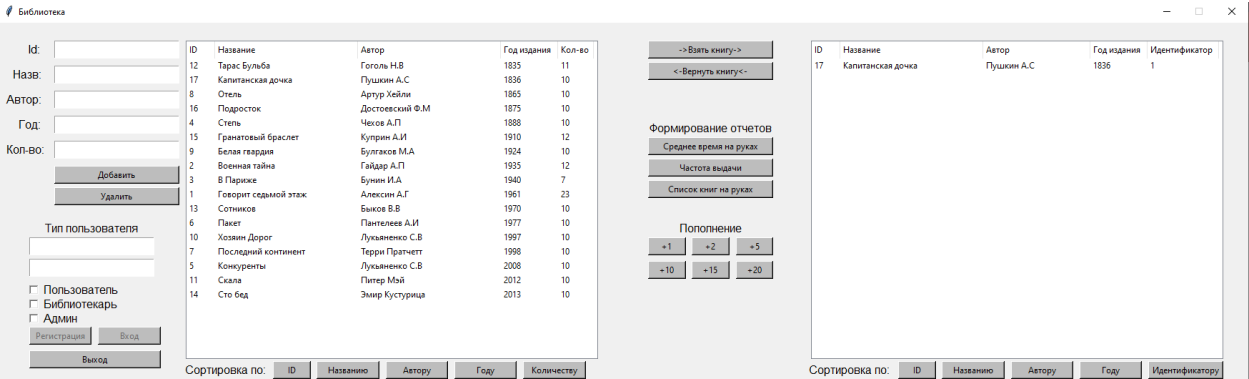


Рис. 16 – Сортировка содержимого списка по году издания

3.3. Вывод к третьей главе

В результате работы над проектной частью был спроектирован пользовательский интерфейс проекта.

4. Тестирование приложения

4.1. Тестирование программного приложения

Целью проведения испытаний является:

- проверка работоспособности функций программного продукта;
- проверка надежности функционирования программного продукта;
- проверка соответствия заявленным характеристикам и требованиям, изложенным в техническом задании;
- проверка соответствия фактического результата работы программы с теоретическим;

Входные данные	Ожидаемый результат	Фактический результат	Статус теста
Пустые поля ввода	Сообщение о том, что все поля	Сообщение о том, что все поля	Пройден

	должны быть заполнены (рис 1)	должны быть заполнены (рис 1)	
1, Сотников, Быков В.В, 1970, 7	Добавление информации о книге в список	Добавление информации о книге в список	Пройден
2, Белая гвардия, Булгаков М.А, 1924, 5	Добавление информации о книге в список	Добавление информации о книге в список	Пройден
Попытка удаления книги, не осуществив выбор элемента из списка	Сообщение о том, что книга не выбрана	Сообщение о том, что книга не выбрана	Пройден
3, Конкуренты, Лукияненко С.В, 2008, 2	Добавление информации о книге в список	Добавление информации о книге в список	Пройден
15, Гранатовый браслет, Куприн А.И, 1910, 4	Сообщение о том, что введенный id уже содержится в библиотеке (рис 2)	Сообщение о том, что введенный id уже содержится в библиотеке (рис 2)	Пройден
Выбор книги для удаления	Удаление выбранной книги	Удаление выбранной книги	Пройден
Три, Гранатовый браслет, Куприн А.И, 1910, 4	Добавления информации не происходит, т.к. id указан неверно	Добавления информации не происходит, т.к. id указан неверно	Пройден
Попытка авторизации в данными логином: admin и паролем: admin	Сообщение о том, что пользователя с такими данными не существует	Сообщение о том, что пользователя с такими данными не существует	Пройден
17, Капитанская дочка, Пушкин А.С, 1836, 10	Добавление информации о книге в список	Добавление информации о книге в список	Пройден
Попытка авторизации с корректно введенными данными	Успешная авторизация пользователя	Успешная авторизация пользователя	Пройден
Попытка выдачи	Сообщение о	Сообщение о	Пройден

книги, не осуществив выбор элемента из списка	том, что книга не выбрана	том, что книга не выбрана	
18, Степь, Чехов А.П., 1888, две	Сообщение о том, что количество экземпляров должно быть указано числом	Сообщение о том, что количество экземпляров должно быть указано числом	Пройден

### Демонстрация некоторых ошибок

Библиотека

Id:

Назв:

Автор:

Год:

Кол-во:

Добавить

Удалить

Тип пользователя

☐ Пользователь

☐ Библиотекарь

☐ Админ

Регистрация Вход

Выход

ID	Название	Автор	Год издания	Кол-во
1	Говорит седьмой этаж	Алексин А.Г	1961	23
2	Военная тайна	Гайдар А.П	1935	12
3	В Париже	Бунин И.А	1940	7
4	Степь		1888	10
5	Конкур		2008	10
6	Пакет		1977	10
7	Послед		1998	10
8	Отель		1865	10
9	Белая		1924	10
10	Хозяин		1997	10
11	Скала		2012	10
12	Тарас Бульба	Гоголь Н.В	1835	11
13	Сотников	Быков В.В	1970	10
14	Сто бед	Эмир Кустурица	2013	10
15	Гранатовый браслет	Куприн А.И	1910	12
16	Подросток	Достоевский Ф.М	1875	10
17	Капитанская дочка	Пушкин А.С	1836	11

Сортировка по: ID Названию Автору Году Количеству

Рис. 17 – Ошибка при не заполнении требуемых полей



Id: 15

Назв: Гранаторый браслет

Автор: Куприн А.И.

Год: 1910

Кол-во: 4

Добавить

Удалить

Тип пользователя

☐ Пользователь

☐ Библиотекарь

☐ Админ

Регистрация Вход

Выход

ID	Название	Автор	Год издания	Кол-во
1	Говорит седьмой этаж	Алексин А.Г	1961	23
2	Военная тайна	Гайдар А.П	1935	12
3	В Париже	Бунин И.А	1940	7
4	Степь	Чехов А.П	1888	10
5		Чехов С.В	2008	10
6		Чехов А.И	1977	10
7		Чехов А.И	1998	10
8		Чехов А.И	1865	10
9		Чехов А.И	1924	10
10		Чехов С.В	1997	10
11		Чехов А.И	2012	10
12	Тарас Бульба	Гоголь Н.В	1835	11
13	Сотников	Быков В.В	1970	10
14	Сто бед	Эмир Кустурица	2013	10
15	Гранаторый браслет	Куприн А.И	1910	12
16	Подросток	Достоевский Ф.М	1875	10
17	Капитанская дочка	Пушкин А.С	1836	11

Сортировка по: ID Названию Автору Году Количеству

Рис. 18 – Ошибка при добавлении существующего id

Тестовые запуски показали, что программное приложение:

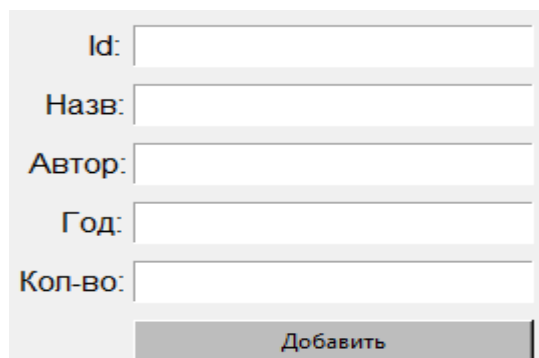
- Работоспособно;
- Устойчиво к ошибкам пользователя;
- Интуитивно понятно, благодаря простому интерфейсу программы;
- Приложение выполнено полностью в соответствии с техническим заданием.

## 5. Инструкция

### 5.1. Инструкция для пользователя

#### Добавление книги:

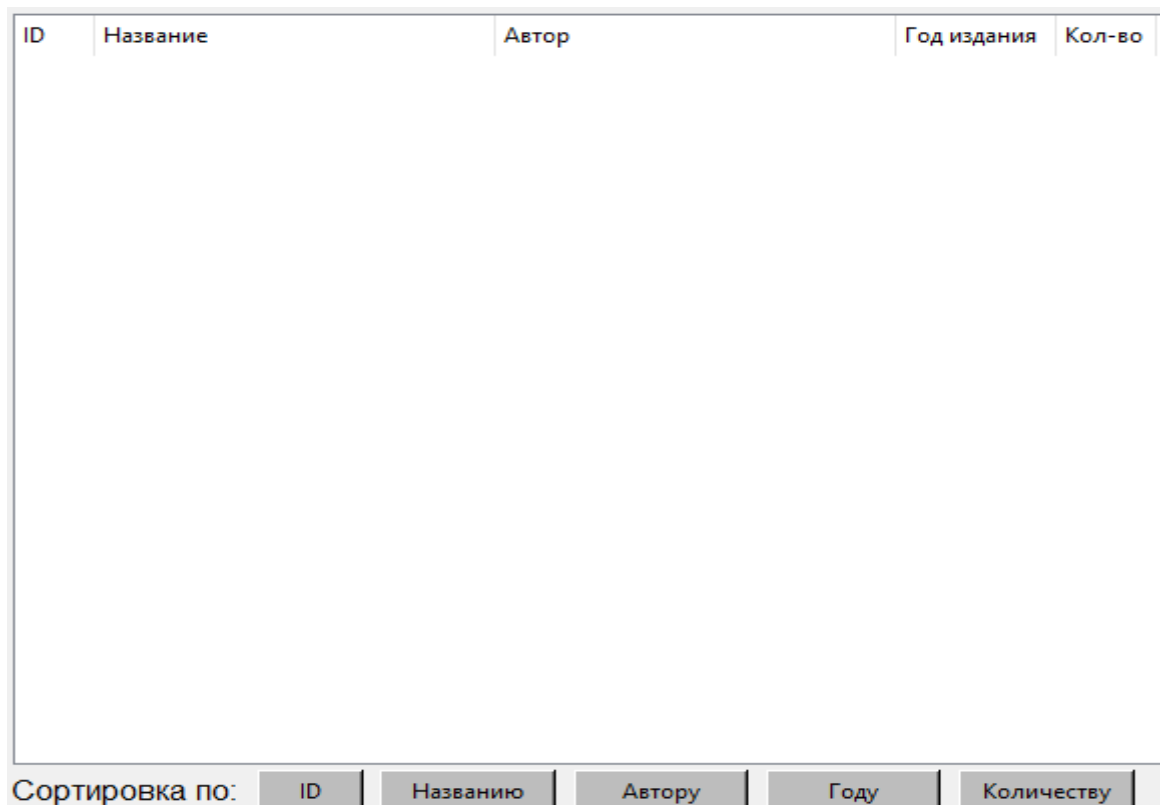
1. Заполнить поля необходимые для добавления. Поля которые требуется заполнять показаны на рисунке 19.



A form for adding a book. It contains five input fields with labels: 'Id:', 'Назв:', 'Автор:', 'Год:', and 'Кол-во:'. Below the fields is a button labeled 'Добавить'.

Рис. 19 — Поля заполнения для добавления книги

2. Нажать на кнопку «Добавить». После добавления книги, она будет выведена в список, находящийся в левой части окна, что изображено на рисунке 20.



ID	Название	Автор	Год издания	Кол-во
----	----------	-------	-------------	--------

Сортировка по: ID Названию Автору Году Количеству

Рис. 20 — Окно вывода списка книг

### Удаление книги:

1. Выбрать из списка книгу, которую необходимо удалить, как показано на рисунке 21.

ID	Название	Автор	Год издания	Кол-во
1	Говорит седьмой этаж	Алексин А.Г	1961	23
2	Военная тайна	Гайдар А.П	1935	12
3	В Париже	Бунин И.А	1940	7
4	Степь	Чехов А.П	1888	10
5	Конкуренты	Лукьяненко С.В	2008	10
6	Пакет	Пантелеев А.И	1977	10
7	Последний континент	Терри Пратчетт	1998	10
8	Отель	Артур Хейли	1865	10
9	Белая гвардия	Булгаков М.А	1924	10
10	Хозяин Дорог	Лукьяненко С.В	1997	10
11	Скала	Питер Мэй	2012	10
12	Тарас Бульба	Гоголь Н.В	1835	11
13	Сотников	Быков В.В	1970	10
14	Сто бед	Эмир Кустурица	2013	10
15	Гранатовый браслет	Куприн А.И	1910	12
16	Подросток	Достоевский Ф.М	1875	10
17	Капитанская дочка	Пушкин А.С	1836	10
555	Дневник памяти	Николас Спаркс	1996	3

Сортировка по: ID Названию Автору Году Количеству

Рис. 21 — Выбор книги для удаления

2. Нажать на кнопку «Удалить», которая изображена на рисунке 22.

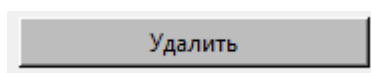


Рис. 22 — Кнопка удаления книги

### Сортировка списка книг:

1. Нажать на одну из кнопок сортировки, которые показаны на рисунке 23.



Рис. 23 — Кнопки сортировки списка книг

## Формирование отчета о среднем времени нахождения книг на руках:

1. Нажать на кнопку «Среднее время на руках», которая показана на рисунке 24, после чего в правом окне программы будет выведен список с результатами, как на рисунке 25.

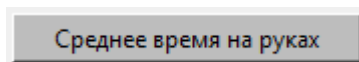


Рис. 24— Кнопка формирования отчета о среднем времени нахождения на руках

ID	Название	Автор	Год издания	Среднее время
1	Говорит седьмой этаж	Алексин А.Г	1961	0.25 дней
2	Военная тайна	Гайдар А.П	1935	0.0 дней
3	В Париже	Бунин И.А	1940	0.0 дней
4	Степь	Чехов А.П	1888	0.0 дней
5	Конкуренты	Лукьяненко С.В	2008	0.0 дней
6	Пакет	Пантелеев А.И	1977	0.0 дней
7	Последний континент	Терри Пратчетт	1998	0.11 дней
8	Отель	Артур Хейли	1865	0.0 дней
9	Белая гвардия	Булгаков М.А	1924	0.0 дней
10	Хозяин Дорог	Лукьяненко С.В	1997	0.0 дней
11	Скала	Питер Мэй	2012	0.0 дней
12	Тарас Бульба	Гоголь Н.В	1835	0.0 дней
13	Сотников	Быков В.В	1970	0.0 дней
14	Сто бед	Эмир Кустурица	2013	0.0 дней
15	Гранатовый браслет	Куприн А.И	1910	0.0 дней
16	Подросток	Достоевский Ф.М	1875	0.4 дней
17	Капитанская дочка	Пушкин А.С	1836	0.0 дней
555	Дневник памяти	Николас Спаркс	1996	0

Сортировка по: ID    Названию    Автору    Году    Времени

Рис. 25— Правое окно программы с результатами запроса

## Формирование отчета о частоте выдачи книг:

1. Нажать на кнопку «Частота выдачи», которая показана на рисунке 26, после чего в правом окне программы будет выведен список с результатами.

Частота выдачи

Рис. 26 — Кнопка формирования отчета о частоте выдачи книг

- После заполнения списка, он будет выведен в список, находящийся в правой части окна, что показано на рисунке 5.9.

ID	Название	Автор	Год издания	Частота выдачи
1	Говорит седьмой этаж	Алексин А.Г	1961	0.53 takes/days
2	Военная тайна	Гайдар А.П	1935	1.4 takes/days
3	В Париже	Бунин И.А	1940	0.71 takes/days
4	Степь	Чехов А.П	1888	0.0 takes/days
5	Конкуренты	Лукьяненко С.В	2008	0.44 takes/days
6	Пакет	Пантелеев А.И	1977	0.0 takes/days
7	Последний континент	Терри Пратчетт	1998	0.0 takes/days
8	Отель	Артур Хейли	1865	0.0 takes/days
9	Белая гвардия	Булгаков М.А	1924	0.0 takes/days
10	Хозяин Дорог	Лукьяненко С.В	1997	0.0 takes/days
11	Скала	Питер Мэй	2012	0.0 takes/days
12	Тарас Бульба	Гоголь Н.В	1835	0.14 takes/days
13	Сотников	Быков В.В	1970	0.0 takes/days
14	Сто бед	Эмир Кустурица	2013	0.0 takes/days
15	Гранатовый браслет	Куприн А.И	1910	0.11 takes/days
16	Подросток	Достоевский Ф.М	1875	0.56 takes/days
17	Капитанская дочка	Пушкин А.С	1836	0.0 takes/days
555	Дневник памяти	Николас Спаркс	1996	0

Сортировка по: ID Названию Автору Году Частоте

Рис. 27 — Правое окно программы с результатами запроса

### Формирование отчета о списке книг, находящихся на руках:

- Нажать на кнопку «Список книг на руках», которая показана на рисунке 28, после чего в правом окне программы будет выведен список с результатами.

Список книг на руках

Рис. 28 — Кнопка формирования отчета о частоте выдачи книг

- После заполнения списка, он будет выведен в список, находящийся в правой части окна, что показано на рисунке 29.

ID	Название	Автор	Год издания	Идентификатор
17	Капитанская дочка	Пушкин А.С	1836	1

Сортировка по:

Рис. 29 — Правое окно программы с результатами запроса

## 6. Заключение

В результате выполнения курсовой работы мы имеем протестированное и исправно работающее программное приложение мониторинга библиотечного фонда.

Реализованное программное приложение имеет следующие функциональные возможности:

- Добавления информации о книге;
- Удаление информации о книге;
- Сортировка книг в библиотеке;
- Реализована система для контроля ввода;
- Формирование отчета о среднем времени пребывания книги на руках;
- Формирование отчета о частоте выдачи книги;
- Формирование отчета о выданных книгах;
- Возможность регистрации;
- Возможность авторизации как администратор, пользователь, работник библиотеки;

В процессе работы над курсовой работой были закреплены навыки:

- Анализа предметной области;
- Работы с базой данных;
- Разработки на языке Python;
- Проектирования графического интерфейса пользователя с помощью стандартной библиотеки Tkinter;
- Тестирования программного продукта;
- Подготовки документации.

## 7.Список используемых источников

Нормативные документы:

1. ГОСТ 7.32-2017 Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления. – М.: Стандартинформ, 2017. – 32 с.;
2. О введении в действие Инструкции по организации и проведению курсового проектирования. – М.: РТУ МИРЭА, Приказ №1325 от 05.10.2018. – 17 с.;

Бумажные источники:

1. Mishra R.K., Raman S.R. Introduction to PySpark SQL. - 978-1-4842-4334-3 изд. - California: Apress, 2019.

Электронные ресурсы:

1. Руководство по языку программирования Python [Электронный ресурс]. – URL: <https://metanit.com/python/tutorial/>
2. Руководство по SQL (полная версия) [Электронный ресурс]. – URL: <https://proselyte.net/tutorials/sql/>
3. Руководство по Tkinter [Электронный ресурс]. – URL: <https://docs.python.org/3/library/tkinter.html>
4. Построение блок-схем [Электронный ресурс]. – URL: <https://www.draw.io>



## 8. Приложение

Файл database.py

```
import sqlite3
```

```
import datetime
```

```
from datetime import datetime
```

```
databaseName = 'dataBase.db'
```

```
def create_tables():
```

```
    connect = sqlite3.connect(databaseName)
```

```
    cursor = connect.cursor()
```

```
    cursor.execute('CREATE TABLE IF NOT EXISTS Library(ID INTEGER,'
```

```
        'Name TEXT,'
```

```
        'Author TEXT,'
```

```
        'Year INTEGER,'
```

```
        'Count INTEGER,'
```

```
        'OnHandsCount INTEGER,'
```

```
        'CountTakes INTEGER,'
```

```
        'AllTime INTEGER,'
```

```
        'middle_time TEXT,'
```

```
        'frequency TEXT,'
```

```
        'add_time TEXT)')
```

```
    cursor.execute('CREATE TABLE IF NOT EXISTS NotInLibrary(ID INTEGER,'
```

```
        'Name TEXT,'
```

```
        'Author TEXT,'
```

```
        'Year INTEGER,'
```

```
        'takeID INTEGER,'
```

```
        'timeTake TEXT,'
```

```
        'takerID INTEGER)')
```

```
    cursor.execute('CREATE TABLE IF NOT EXISTS Users(login TEXT,'
```

```

        'password TEXT,'
        'type TEXT)')
connect.commit()

def fill_libTable():
    connect = sqlite3.connect(databaseName)
    cursor = connect.cursor()
    cursor.execute("SELECT ID,Name,Author,Year,Count FROM Library WHERE Count>0
ORDER BY ID")
    books = cursor.fetchall()
    return books

def fill_onHandTableLib(userID, who):
    connect = sqlite3.connect(databaseName)
    cursor = connect.cursor()
    print(who)
    print(userID)
    if who == 1 or who == 2:
        cursor.execute("SELECT ID,Name,Author,Year,takeID FROM NotInLibrary ORDER BY
ID")
        books = cursor.fetchall()
        return books
    elif who == 0:
        cursor.execute("SELECT ID,Name,Author,Year,takeID FROM NotInLibrary WHERE
takerID={0} ORDER BY ID".
                        format(userID))
        books = cursor.fetchall()
        return books

```

```
def fill_onHandTableUser(userID):
```

```
    connect = sqlite3.connect(databaseName)
```

```
    cursor = connect.cursor()
```

```
    cursor.execute("SELECT ID,Name,Author,Year,takeID FROM NotInLibrary WHERE  
takerID={0} ORDER BY ID".format(userID))
```

```
    books = cursor.fetchall()
```

```
    return books
```

```
def fill_middle():
```

```
    connect = sqlite3.connect(databaseName)
```

```
    cursor = connect.cursor()
```

```
    cursor.execute("SELECT ID,Name,Author,Year,middle_time FROM Library ORDER BY  
ID")
```

```
    books = cursor.fetchall()
```

```
    return books
```

```
def fill_frequency():
```

```
    connect = sqlite3.connect(databaseName)
```

```
    cursor = connect.cursor()
```

```
    cursor.execute("SELECT ID,Name,Author,Year,frequency FROM Library ORDER BY ID")
```

```
    books = cursor.fetchall()
```

```
    return books
```

```
def sort1(byWhat):
```

```
    connect = sqlite3.connect(databaseName)
```

```
    cursor = connect.cursor()
```

```
    cursor.execute("SELECT ID,Name,Author,Year,Count FROM Library WHERE Count>0  
ORDER BY " + str(byWhat))
```

```
    return cursor.fetchall()
```

```

def sort2(byWhat, who, userID, table):
    connect = sqlite3.connect(databaseName)
    cursor = connect.cursor()
    if table == 0:
        if who == 1 or who == 2:
            cursor.execute("SELECT ID,Name,Author,Year,takeID FROM NotInLibrary ORDER
BY " + str(byWhat))
        elif who == 0:
            cursor.execute("SELECT ID,Name,Author,Year,takeID FROM NotInLibrary WHERE
takerID={0} ORDER BY {1}".
                            format(userID, byWhat))
        return cursor.fetchall()
    elif table == 1:
        if byWhat == "takeID":
            byWhat = "middle_time"
        if who == 1 or who == 2:
            cursor.execute("SELECT ID,Name,Author,Year,middle_time FROM Library ORDER
BY " + str(byWhat))
        elif who == 0:
            cursor.execute("SELECT ID,Name,Author,Year,middle_time FROM Library WHERE
takerID={0} ORDER BY {1}".
                            format(userID, byWhat))
        return cursor.fetchall()
    else:
        if byWhat == "takeID":
            byWhat = "frequency"
        if who == 1 or who == 2:
            cursor.execute("SELECT ID,Name,Author,Year,frequency FROM Library ORDER BY "
+ str(byWhat))
        elif who == 0:

```

```

        cursor.execute("SELECT ID,Name,Author,Year,frequency FROM Library WHERE
takerID={0} ORDER BY {1}").

```

```

        format(userID, byWhat))

```

```

    return cursor.fetchall()

```

```

def add_countBooks(ID, count):

```

```

    try:

```

```

        connect = sqlite3.connect(databaseName)

```

```

        cursor = connect.cursor()

```

```

        cursor.execute("UPDATE Library SET Count=Count+{0} WHERE ID={1}".format(count,
ID))

```

```

        connect.commit()

```

```

    except Exception as e:

```

```

        print(e)

```

```

def add_to_database(data):

```

```

    try:

```

```

        connect = sqlite3.connect(databaseName)

```

```

        cursor = connect.cursor()

```

```

        cursor.execute("INSERT INTO library VALUES (?, ?, ?, ?, ?, 0, 0, 0, '0', '0', '{0}')"

```

```

            .format(datetime.now().strftime('%y-%m-%d')), data)

```

```

        connect.commit()

```

```

    except Exception as e:

```

```

        print(e)

```

```

def del_from_database(ID):

```

```

    try:

```

```

        connect = sqlite3.connect(databaseName)

```

```

        cursor = connect.cursor()

        cursor.execute("DELETE FROM library WHERE ID=" + str(ID))

        connect.commit()

    except Exception as e:

        print(e)


def check_id(ID):

    try:

        connect = sqlite3.connect(databaseName)

        cursor = connect.cursor()

        cursor.execute("SELECT ID FROM Library WHERE ID=" + str(ID))

        contain = cursor.fetchall()[0]

        print(contain)

        return False

    except IndexError:

        return True


def take_book(ID, takeID):

    try:

        connect = sqlite3.connect(databaseName)

        cursor = connect.cursor()

        cursor.execute("SELECT timeTake FROM NotInLibrary WHERE takeID=" + str(takeID))

        time = cursor.fetchall()[0][0]

        date_format = '%y-%m-%d'

        time = datetime.strptime(time, date_format)

        now = datetime.strptime(datetime.now().strftime('%y-%m-%d'), date_format)

        res = now - time

        res = int(res.days)

```

```

        cursor.execute("UPDATE Library SET Count=Count+1,OnHandsCount=OnHandsCount-1,AllTime=AllTime+{0} WHERE ID={1}"

```

```

                .format(res, ID))

```

```

        cursor.execute("DELETE FROM NotInLibrary WHERE ID={0} AND
TakeID={1}".format(ID, takeID))

```

```

        connect.commit()

```

```

    except Exception as e:

```

```

        print(e)

```

```

def give_book(ID, takerID):

```

```

    try:

```

```

        connect = sqlite3.connect(databaseName)

```

```

        cursor = connect.cursor()

```

```

        cursor.execute("SELECT Count FROM Library WHERE ID=" + str(ID))

```

```

        count = int(cursor.fetchall()[0][0])

```

```

        connect.commit()

```

```

        if count > 0:

```

```

            cursor.execute("UPDATE Library SET Count=Count-1,OnHandsCount=OnHandsCount+1,CountTakes=CountTakes+1 "

```

```

                "WHERE ID=" + str(ID))

```

```

            cursor.execute("SELECT ID,Name,Author,Year FROM Library WHERE ID=" + str(ID))

```

```

            data = cursor.fetchall()[0]

```

```

            cursor.execute("INSERT INTO NotInLibrary VALUES (?, ?, ?, ?, {0}, '{1}', {2})"

```

```

                .format(get_max_ID(), datetime.now().strftime('%y-%m-%d'), takerID), data)

```

```

            connect.commit()

```

```

            return count

```

```

        else:

```

```

            cursor.execute("UPDATE Library SET Count=0 WHERE ID=" + str(ID))

```

```

            connect.commit()

```

```

            return count

```

```

    except Exception as e:

```

```
print(e)
```

```
def get_book(ID):  
    connect = sqlite3.connect(databaseName)  
    cursor = connect.cursor()  
    cursor.execute("SELECT ID,Name,Author,Year,Count FROM Library WHERE ID=" +  
str(ID))  
    book = cursor.fetchall()[0]  
    return book
```

```
def get_book_onHand(ID):  
    connect = sqlite3.connect(databaseName)  
    cursor = connect.cursor()  
    cursor.execute("SELECT MAX(takeID) FROM NotInLibrary WHERE ID=" + str(ID))  
    maxID = cursor.fetchall()[0][0]  
    cursor.execute("SELECT ID,Name,Author,Year,takeID FROM NotInLibrary WHERE  
takeID=" + str(maxID))  
    book = cursor.fetchall()[0]  
    return book
```

```
def get_max_ID():  
    try:  
        connect = sqlite3.connect(databaseName)  
        cursor = connect.cursor()  
        cursor.execute("SELECT MAX(takeID) FROM NotInLibrary")  
        maxID = int(cursor.fetchall()[0][0])  
        maxID += 1  
        connect.commit()
```



```

        return maxID
    except TypeError:
        return 1

def get_middleTime(ID):
    try:
        connect = sqlite3.connect(databaseName)
        cursor = connect.cursor()
        cursor.execute("SELECT AllTime,CountTakes FROM Library WHERE ID=" + str(ID))
        res = cursor.fetchall()
        time = res[0][0]
        takes = res[0][1]
        if takes == 0:
            cursor.execute("UPDATE Library SET middle_time='{0}' WHERE ID={1}".format(0,
ID))
            connect.commit()
            return 0
        middle = time / takes
        last_num = str(middle)[len(str(middle)) - 1]
        if middle == 1 or last_num == '1':
            day = ' день'
        elif 5 > middle > 1 or last_num == '2' or last_num == '2' or last_num == '2':
            day = ' дня'
        else:
            day = ' дней'
        cursor.execute(
            "UPDATE Library SET middle_time='{0}' WHERE ID={1}".format(str(round(time /
takes, 2)) + day, ID))
        connect.commit()
        return round(time / takes, 2)

```

```
except Exception as e:
```

```
    print(e)
```

```
def get_frequency(ID):
```

```
    try:
```

```
        connect = sqlite3.connect(databaseName)
```

```
        cursor = connect.cursor()
```

```
        cursor.execute("SELECT CountTakes,add_time FROM Library WHERE ID=" + str(ID))
```

```
        res = cursor.fetchall()
```

```
        takes = res[0][0]
```

```
        time = res[0][1]
```

```
        date_format = '%y-%m-%d'
```

```
        time = datetime.strptime(time, date_format)
```

```
        now = datetime.strptime(datetime.now().strftime('%y-%m-%d'), date_format)
```

```
        res = now - time
```

```
        res = int(res.days)
```

```
        freq = str(round(takes / (res + 1), 2))
```

```
        freq += " takes/days"
```

```
        cursor.execute("UPDATE Library SET frequency='{0}' WHERE ID={1}".format(freq, ID))
```

```
        connect.commit()
```

```
    except Exception as e:
```

```
        print(e)
```

```
def check_user(login, password):
```

```
    try:
```

```
        connect = sqlite3.connect(databaseName)
```

```
        cursor = connect.cursor()
```

```
        cursor.execute("SELECT login,password,type FROM Users WHERE login=" + str(login))
```

```

res = cursor.fetchall()
Log = res[0][0]
Pass = res[0][1]
Type = res[0][2]
if Log == login and Pass == password:
    return Type
else:
    return "5"
except IndexError as e:
    return False

def reg_user(login, password, Type):
    try:
        connect = sqlite3.connect(databaseName)
        cursor = connect.cursor()
        cursor.execute("SELECT login FROM Users WHERE login=" + str(login))
        try:
            user = cursor.fetchall()[0][0]
            print(user)
            return False
        except IndexError:
            print('1')
        data = [login, password, Type]
        cursor.execute("INSERT INTO Users VALUES(?,?,?)", data)
        connect.commit()
        return True
    except IndexError:
        pass

```