**МИНИСТЕРСТВО ОБРАЗОВАНИЯ НОВОСИБИРСКОЙ ОБЛАСТИ**

Государственное автономное профессиональное образовательное учреждение Новосибирской области

«Новосибирский колледж печати и информационных технологий»



ОТЧЁТ

по лабораторной работе №2

«Разработка CRUD-приложения для управления

библиотекой книг с использованием Python и

PostgreSQL»

по дисциплине: «Разработка кода информационных систем»

|  |  |
| --- | --- |
| Выполнил:  Студент гр. 9-22, ИСиП:  Андрущук Д. В. | Преподаватель:  Перевалов Г.Г. |

Новосибирск 2024 г

Оглавление

[1. ВВЕДЕНИЕ 3](#_Toc177251811)

[1.1. Цель и задачи 3](#_Toc177251812)

[1.2. Требования к системе: 3](#_Toc177251813)

[2. РАЗРАБОТКА ПРОЕКТА 4](#_Toc177251814)

[2.1. Соединение с базой данных 4](#_Toc177251815)

[2.2. Обработка ошибок 4](#_Toc177251816)

[2.3. Добавление книги 4](#_Toc177251817)

[2.4. Вывод всех книг на экран 5](#_Toc177251818)

[2.5. Удаление книги 5](#_Toc177251819)

[2.6. Поиск книги по ее названию 5](#_Toc177251820)

[2.7. Поиск книги по ее автору 6](#_Toc177251821)

[2.8. Обновление книги по id 6](#_Toc177251822)

[2.9. Вывод меню на экран 7](#_Toc177251833)

[2.10. Исполняемая программа(main) 7](#_Toc177251834)

[3. ПОКАЗ РАБОТЫ ПРОГРАММЫ 8](#_Toc177251835)

[3.1. Добавление книги 8](#_Toc177251836)

[3.3. Удаление книги 11](#_Toc177251837)

[3.4. Поиск по автору 12](#_Toc177251838)

[3.5. Поиск по названию книги 13](#_Toc177251840)

[3.6. Обновление книги по id 13](#_Toc177251841)

[3.7. Выход из программы 14](#_Toc177251842)

[3.8. Обработка ошибок 15](#_Toc177251843)

[4. ВЫВОД 16](#_Toc177251844)

# ВВЕДЕНИЕ

## Цель и задачи

Цель: Разработка CRUD-приложения для управления библиотекой книг с использованием Python и PostgreSQL

Задачи:

* Разработать консольное приложение на Python для управления библиотекой книг.
* Приложение должно взаимодействовать с базой данных PostgreSQL и предоставлять функциональность для выполнения следующих операций:
* Добавление новой книги
* Просмотр списка всех книг
* Поиск книги по названию и автору
* Обновление информации о книге
* Удаление книги

## Требования к системе:

Каждая книга должна иметь следующие поля:

* Идентификатор (автоматически генерируется)
* Название книги
* Автор
* Год издания
* Жанр
* Количество страниц

Приложение должно поддерживать подключение к базе данных PostgreSQL с использованием библиотеки psycopg2.

Реализовать все операции CRUD (создание, чтение, обновление, удаление).

# РАЗРАБОТКА ПРОЕКТА

## Соединение с базой данных

import psycopg2

conn = psycopg2.connect(database="books", user="postgres",

    password="123", host="localhost", port=5432)

cur = conn.cursor()

# cur.execute("CREATE TABLE book (id SERIAL PRIMARY KEY, " +

#     "name VARCHAR(100), author VARCHAR(100), year INTEGER, genre VARCHAR(100), pages INTEGER)")

# conn.commit()

Здесь подключается библиотека котороя работает с Postgres, вводится переменная для соединения и для SQL-запросов. Таблица book сосдается всего 1 раз, при порторном запуске ее не нужно создавать.

## Обработка ошибок

def error(per):

    try:

        per = int(per)

    except ValueError:

        print("Input the number")

        return False

    return True

Эта функция позволяет обработать ошибку, если нужно, а с клавиатуры вводится строка. Таких переменных три: id, год издания, количество страниц.

## Добавление книги

def add():

    book\_name = input("Name:    ")

    book\_author = input("Author:    ")

    while True:

        book\_year = input("Year:    ")

        if (error(book\_year)):

            break

    book\_genre = input("Genre:  ")

    while True:

        book\_pages = input("Pages:  ")

        if (error(book\_pages)):

            break

    cur.execute("INSERT INTO book (name, author, year, genre, pages) VALUES (%s, %s, %s, %s, %s)", (book\_name, book\_author, book\_year, book\_genre, book\_pages))

    conn.commit()

Здесь запрашиваются данные о книге, год издания и количество страниц запрашивается до тех пор пока не будет введено число. Она добавляет в базу данных строку с введенными данными.

## Вывод всех книг на экран

def print\_all():

    print("\nALL BOOKS:\n")

    cur.execute("SELECT \* FROM book")

    for row in cur:

        print(f" Book #{row[0]}\n Name: {row[1]}\n Author: {row[2]}\n Year: {row[3]}\n Genre: {row[4]}\n Pages: {row[5]}\n\n")

Здесь запрашиваются все данные из базы данных и представляются в виде кортежа со строками. Происходит перебор каждой строки и данные вставляются в текст, который выводится на экран.

## Удаление книги

def delete():

    while True:

        id\_book = input("Id:    ")

        if (error(id\_book)):

            break

    cur.execute("DELETE FROM book WHERE id = %s", (id\_book,))

    conn.commit()

Удаление происходит по id, он запрашивается с клавиатуры, идет проверка на числовое значение. Удаляется запись из базы данных с этим id.

## Поиск книги по ее названию

def search\_by\_name():

    book\_name = input("Name:    ")

    print(f"\nBooks named {book\_name}")

    cur.execute("SELECT \* FROM book WHERE name = %s", (book\_name,))

    for row in cur:

        print(f" Book #{row[0]}\n Author: {row[2]}\n Year: {row[3]}\n Genre: {row[4]}\n Pages: {row[5]}\n\n")

Эта функция выбирает из базы данных все записи, где столбец имя такой же как введенное с клавиатуры значение. Выводит на экран соответсятвующие строки.

## Поиск книги по ее автору

def search\_by\_author():

    book\_author = input("Author:    ")

    print(f"\nBooks of {book\_author}")

    cur.execute("SELECT \* FROM book WHERE author = %s", (book\_author,))

    for row in cur:

        print(f" Book #{row[0]}\n Name: {row[1]}\n  Year: {row[3]}\n Genre: {row[4]}\n Pages: {row[5]}\n\n")

Эта функция аналогично выполняет поиск книги по автору.

## Обновление книги по id

def update():

    while True:

        id\_book = input("Id:    ")

        if (error(id\_book)):

            break

    cur.execute("SELECT id FROM book WHERE id = %s", (id\_book,))

    # print(cur.rowcount)

    if (cur.rowcount == 0):

        print("Id is not exists")

        return

   book\_name = input("Name:    ")

    book\_author = input("Author:    ")

    while True:

        book\_year = input("Year:    ")

        if (error(book\_year)):

            break

    book\_genre = input("Genre:  ")

    while True:

        book\_pages = input("Pages:  ")

        if (error(book\_pages)):

            break

    cur.execute("UPDATE book SET name = %s, author = %s, year = %s, genre = %s, pages = %s WHERE id = %s", (book\_name, book\_author, book\_year, book\_genre, book\_pages, id\_book))

    conn.commit()

Здесь проверяется на наличие id в базе данных и проверка на числовое значение. Делает запрос в базу данных на обновление записей, где id такой же как введенное значение.



## Вывод меню на экран

def menu():

    print("--------------------------------------")

    print("1. Add book\n2. Delete book\n3. Search by author\n4. Search by name\n5. Update book by id\n6. Print all books\n7. Exit")

    print("--------------------------------------")

## Исполняемая программа(main)

while True:

    menu()

    choice = input("Enter your choice:  ")

    try:

        choice = int(choice)

    except ValueError:

        print("Incorrect input")

    if (choice == 1):

        add()

    elif (choice == 2):

        delete()

    elif (choice == 3):

        search\_by\_author()

    elif (choice == 4):

        search\_by\_name()

    elif (choice == 5):

        update()

    elif (choice == 6):

        print\_all()

    elif (choice == 7):

        break

    else:

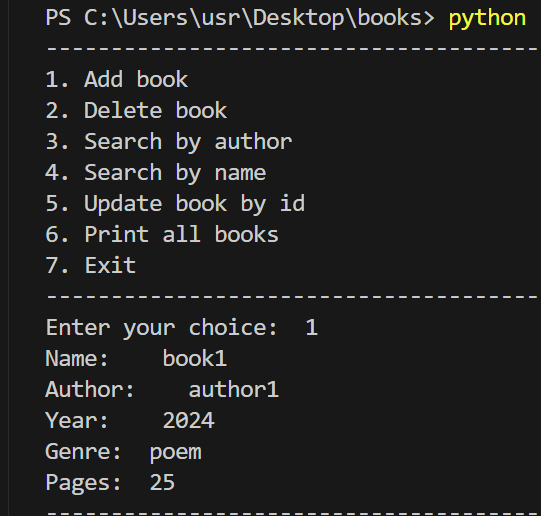
        print("Choice is not exists")

conn.close()

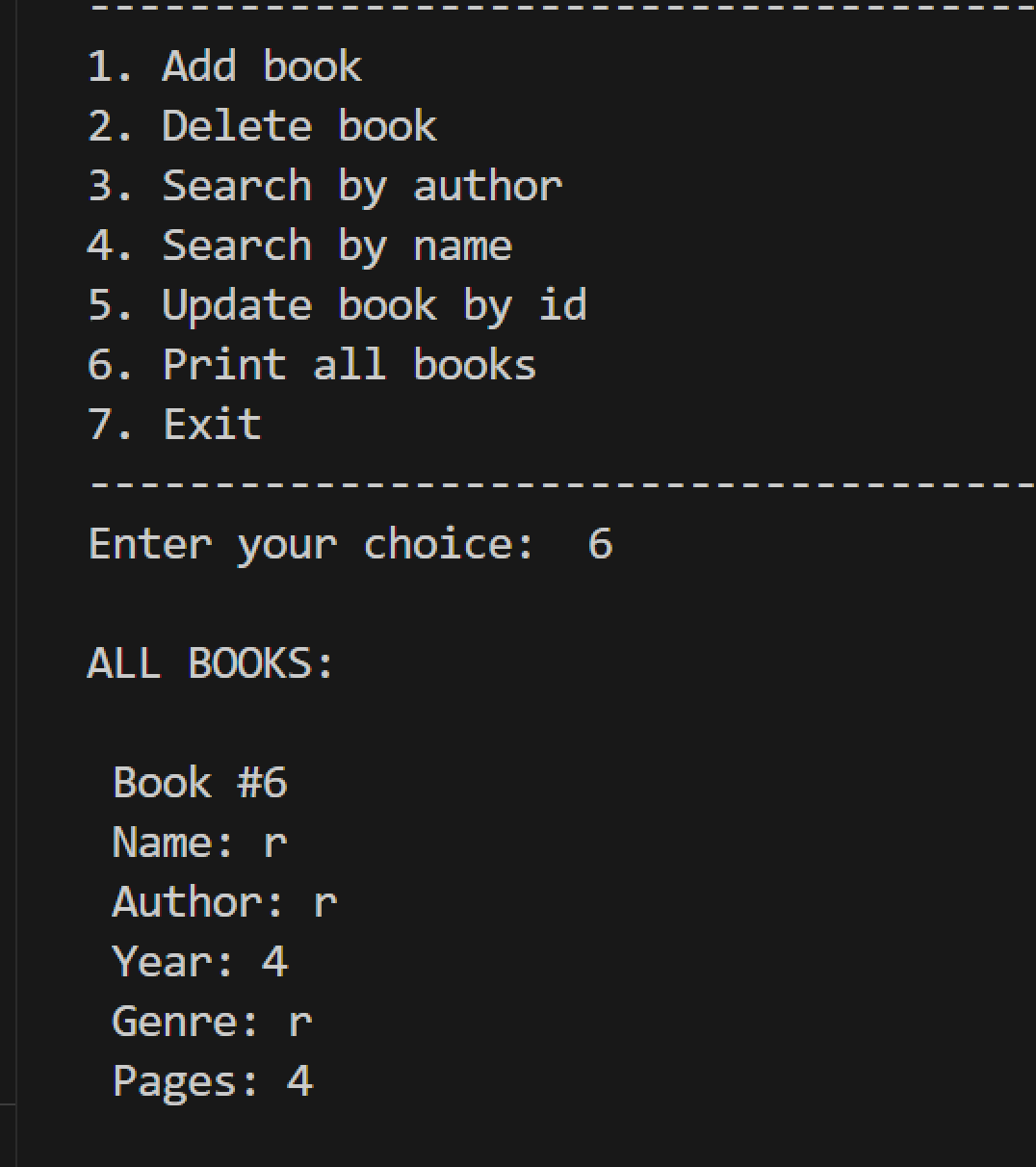
Здесь запрашивается номер операции в соответствии с меню и обрабатывается ошибка введения строки или номера, которого нет в списке.

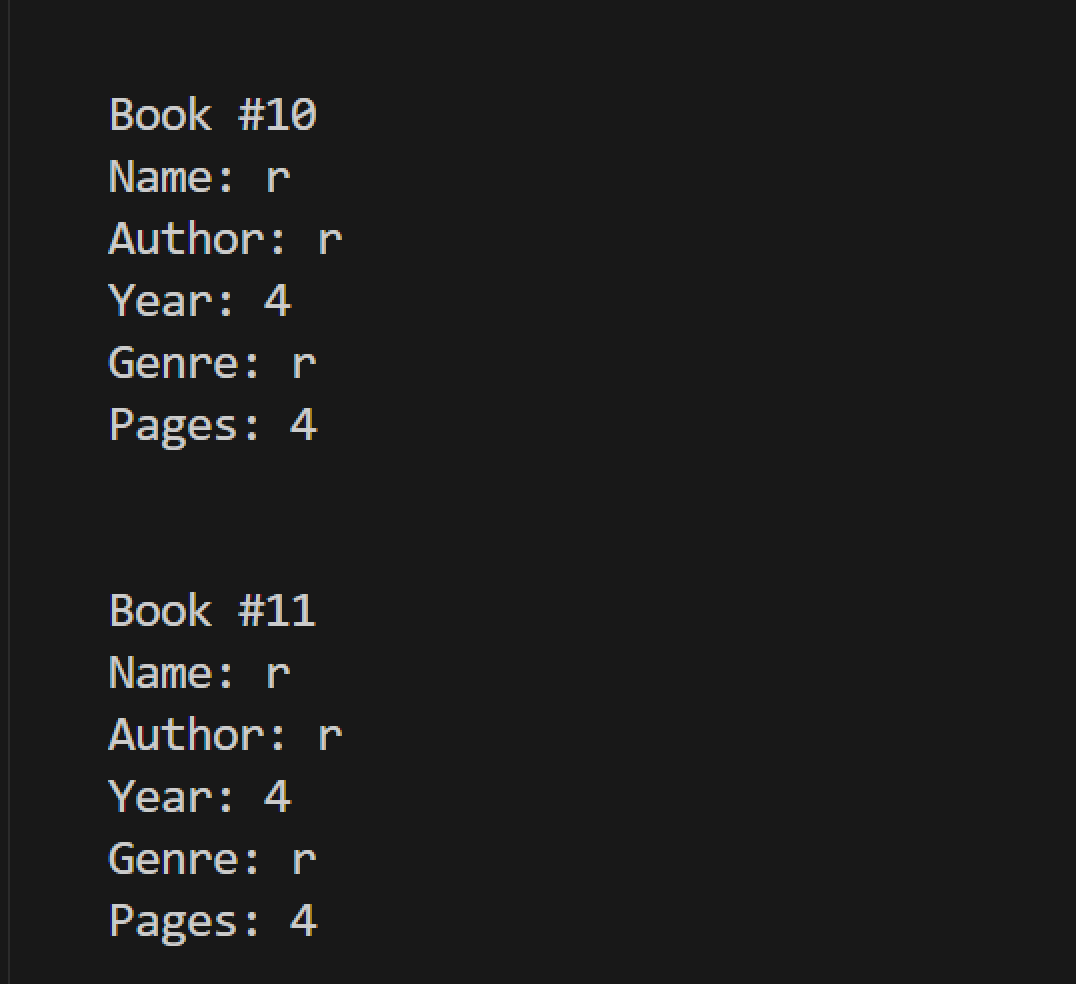
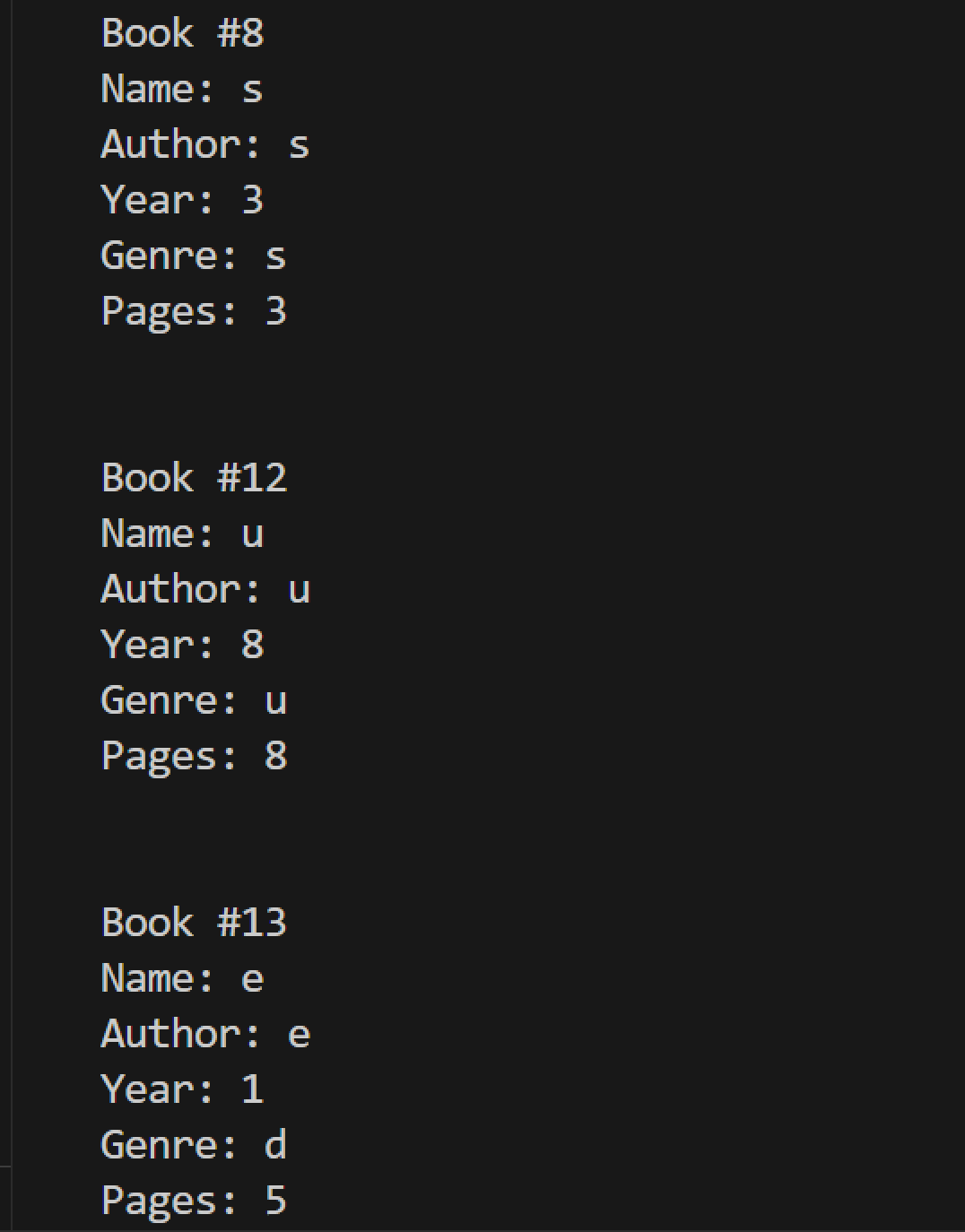
# ПОКАЗ РАБОТЫ ПРОГРАММЫ

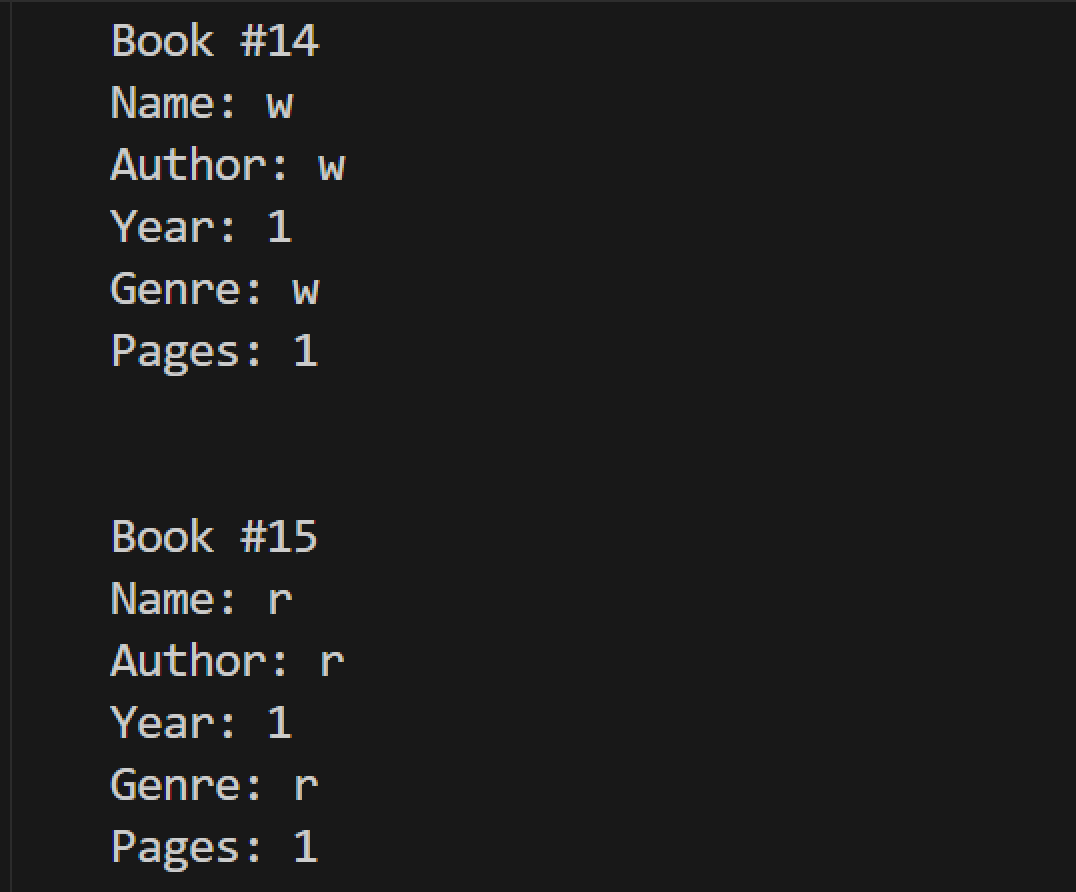
## Добавление книги

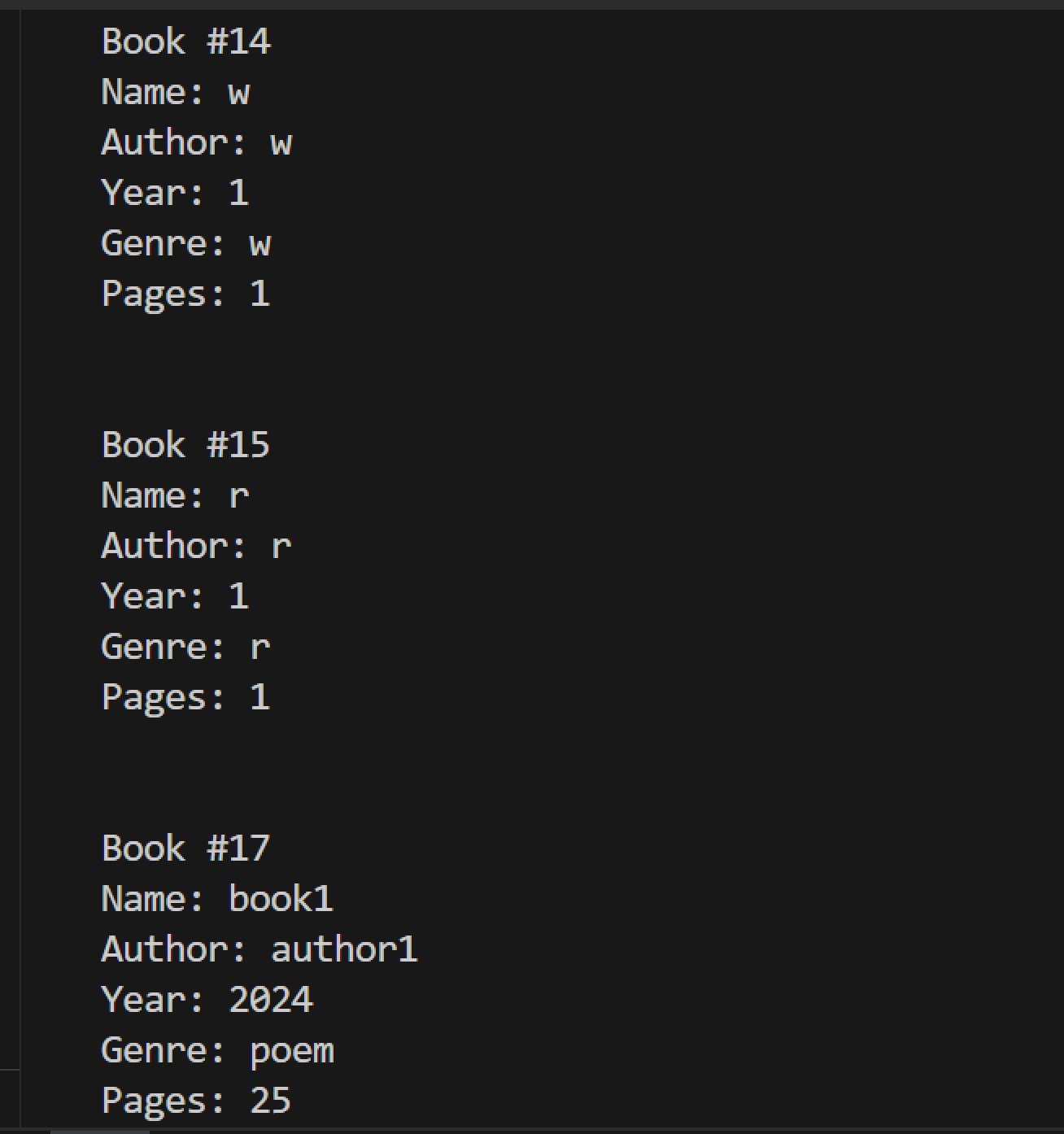


* 1. Вывод всех книг на экран

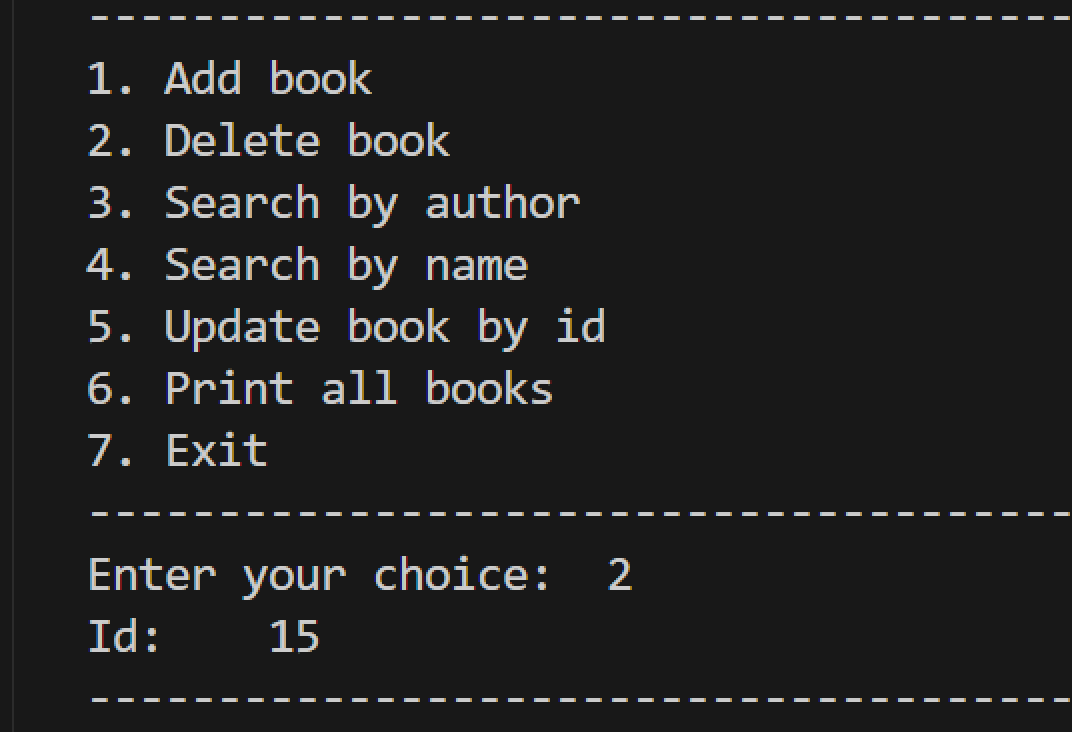


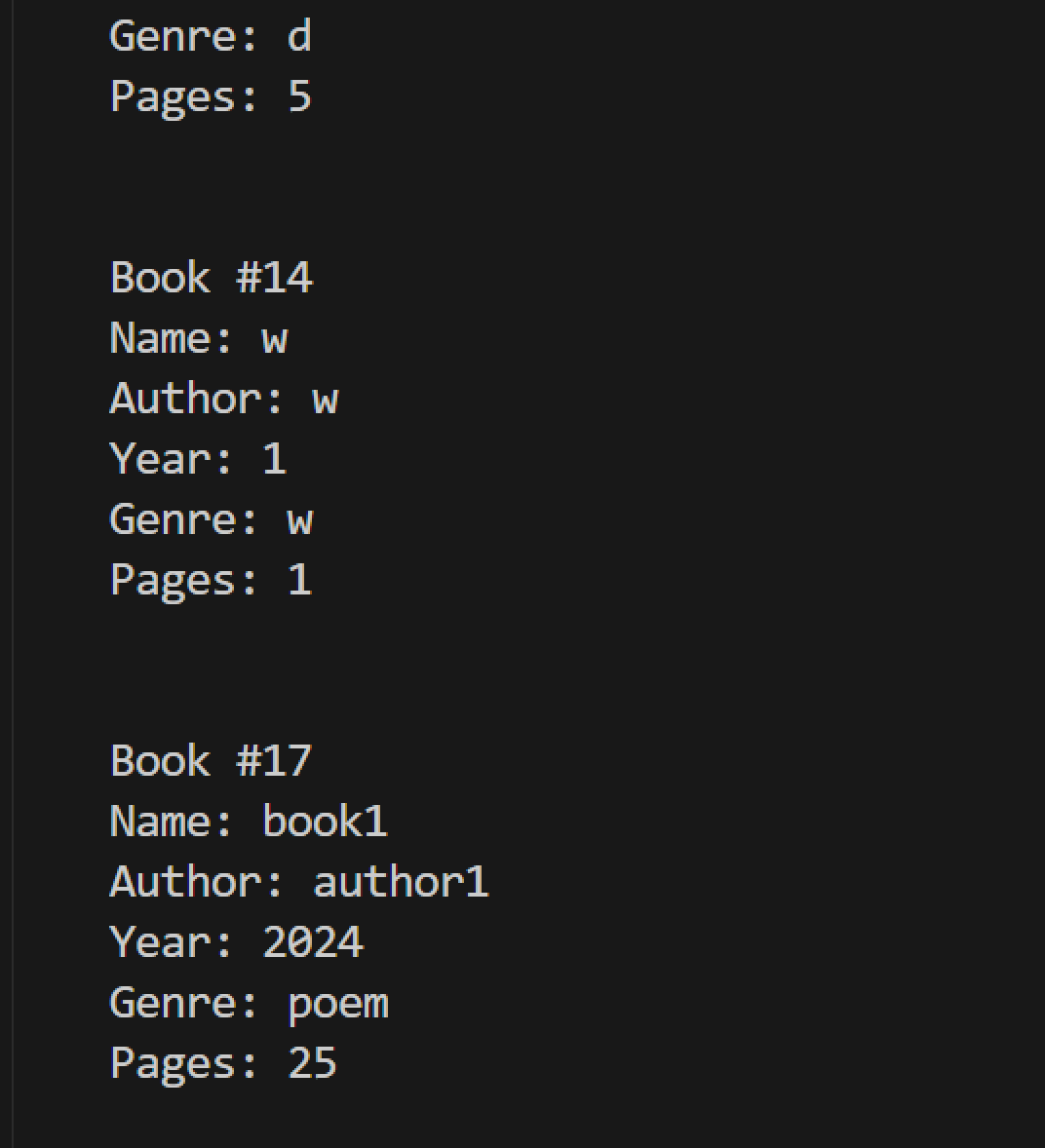




## Удаление книги



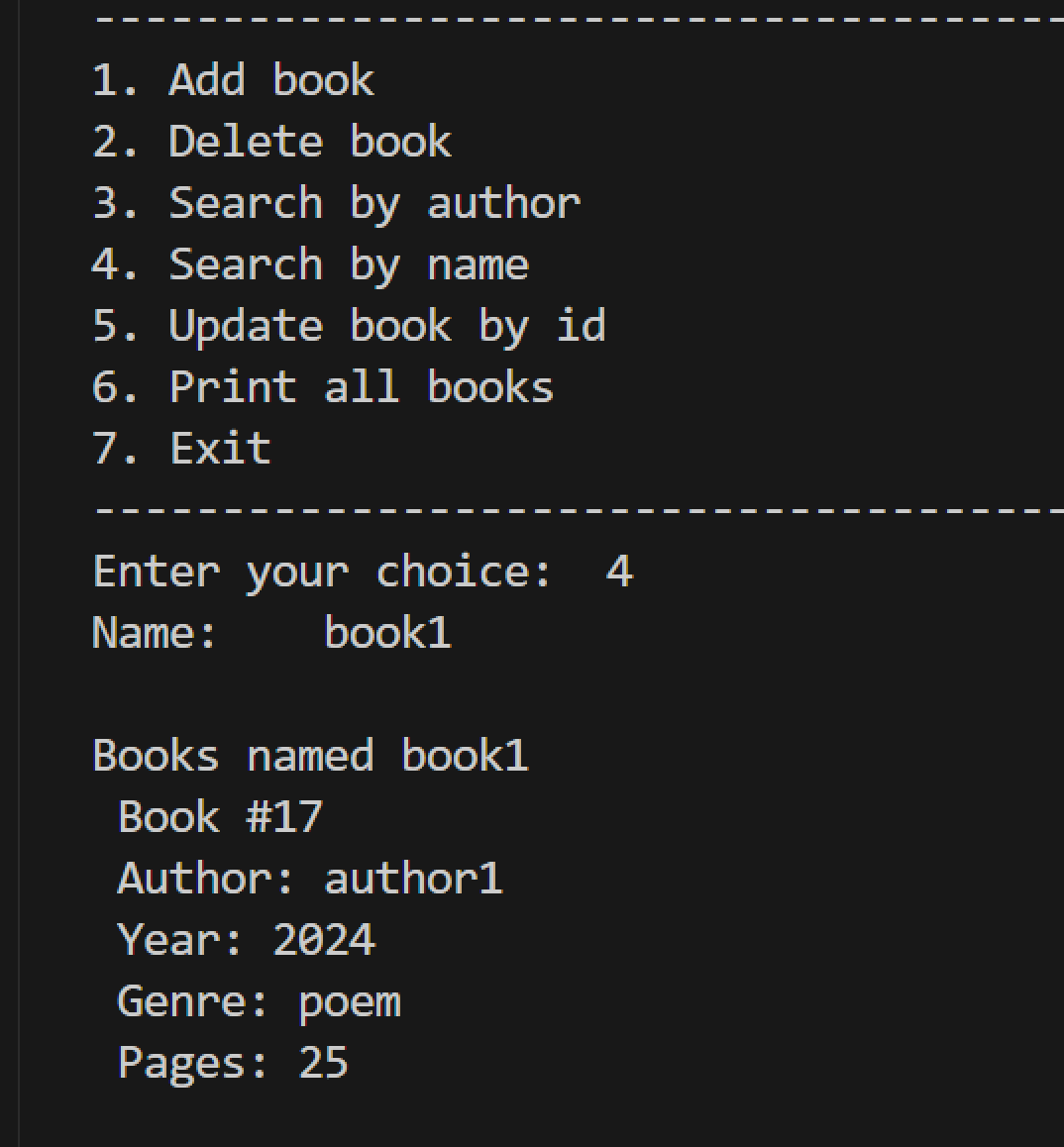
Видно ниже что такой книги больше нет



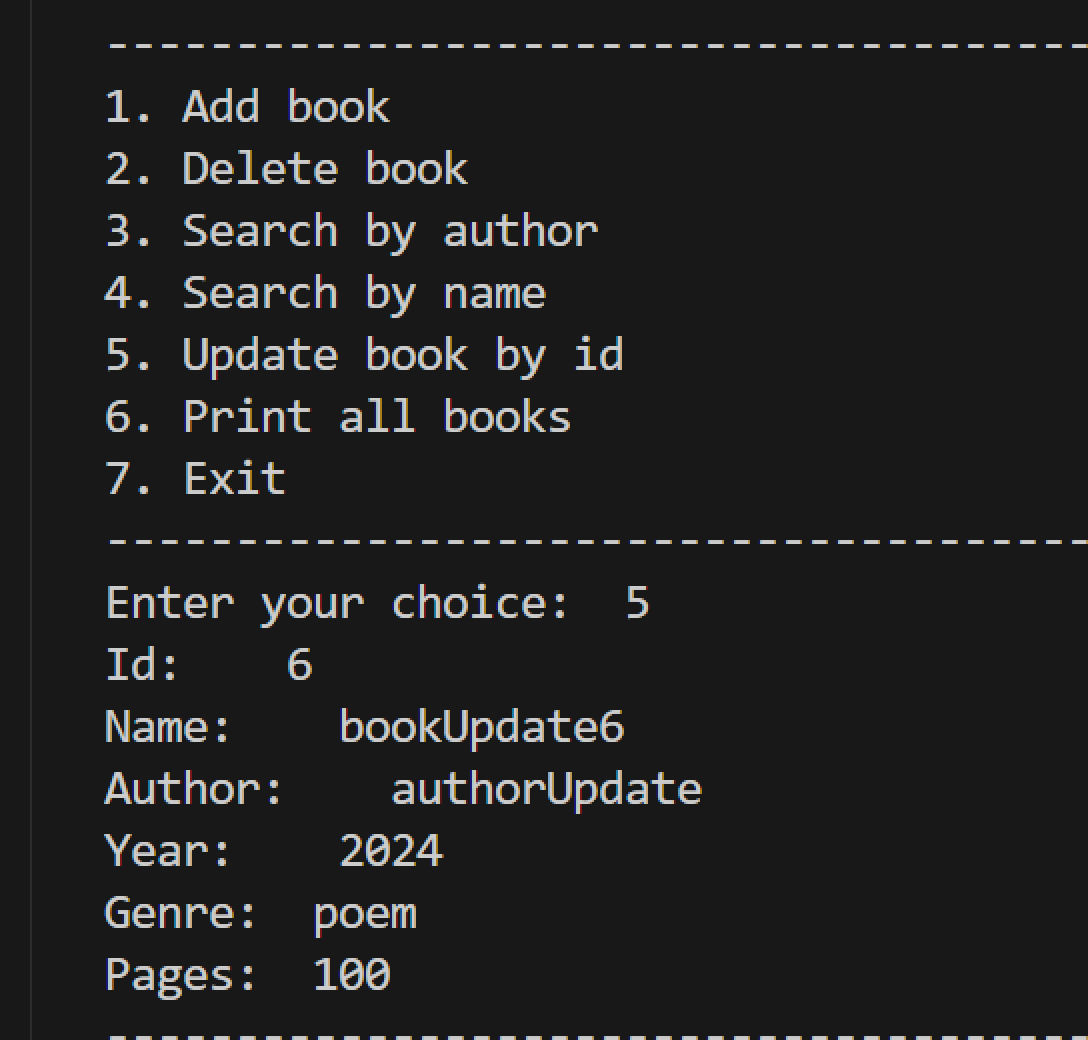
## Поиск по автору

## 

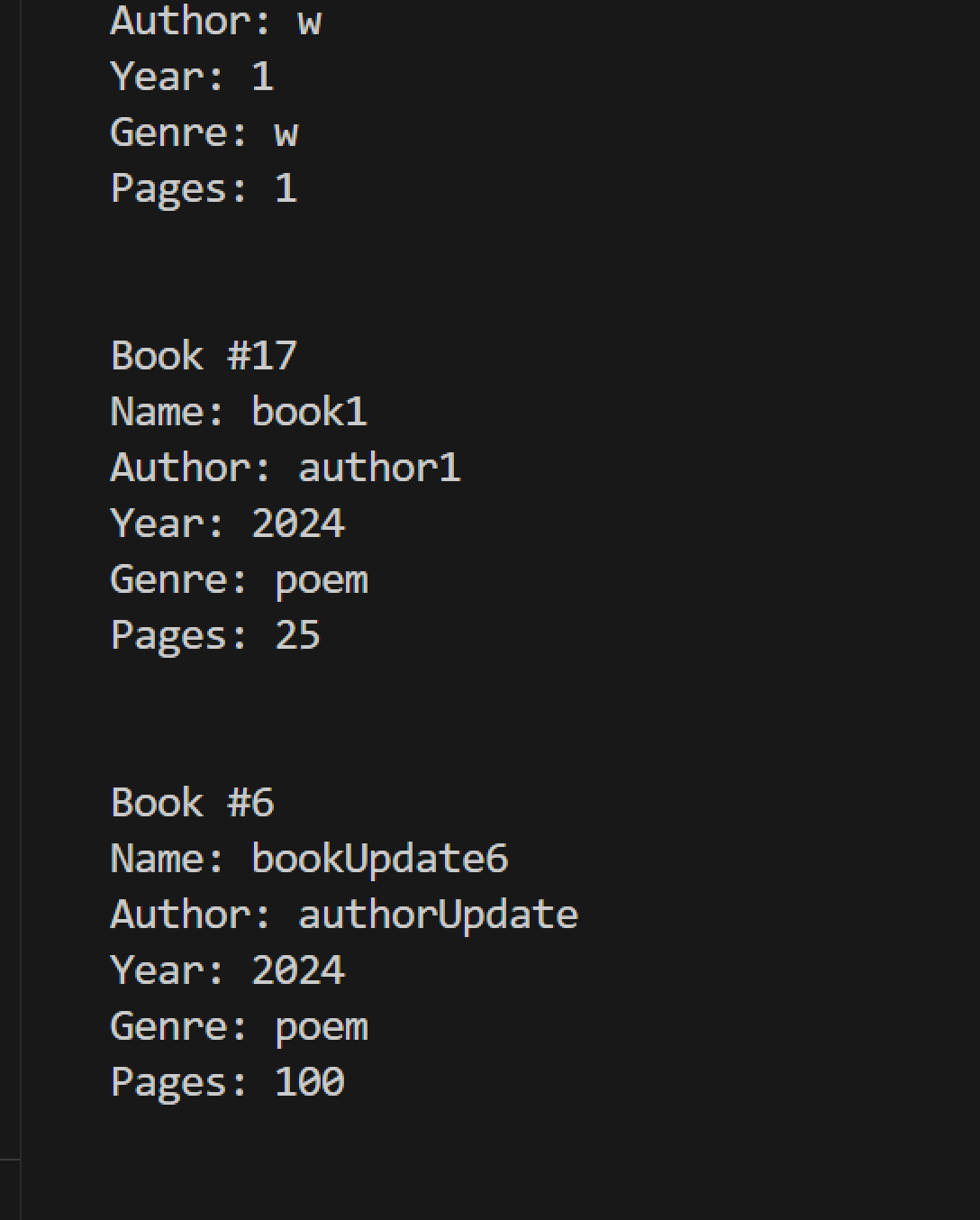
## Поиск по названию книги



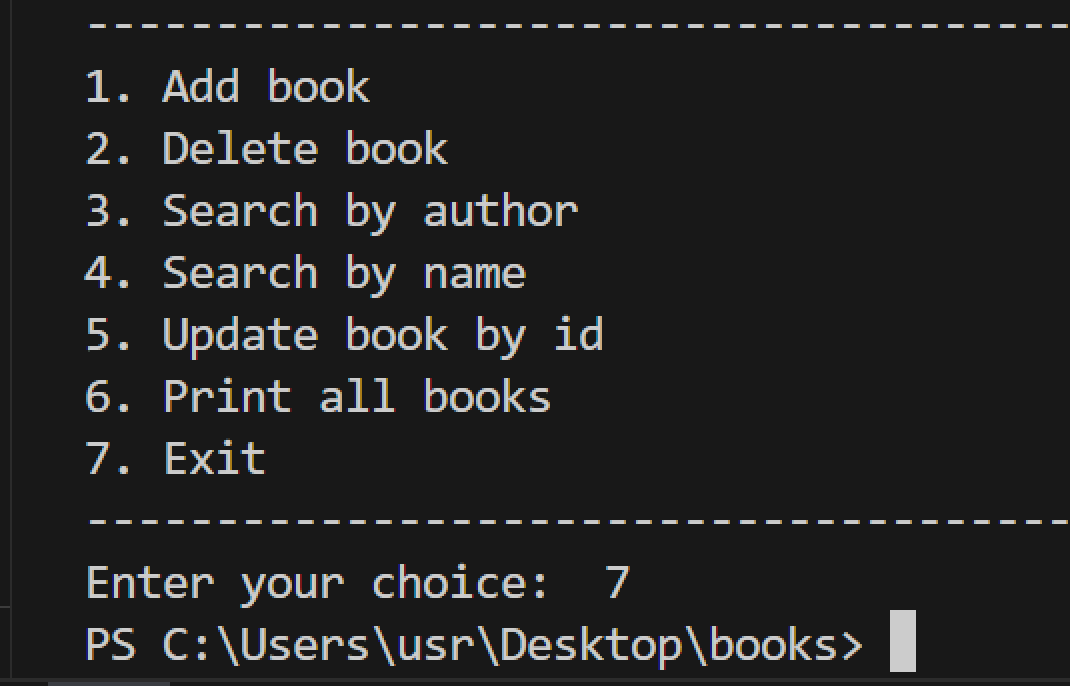
## Обновление книги по id



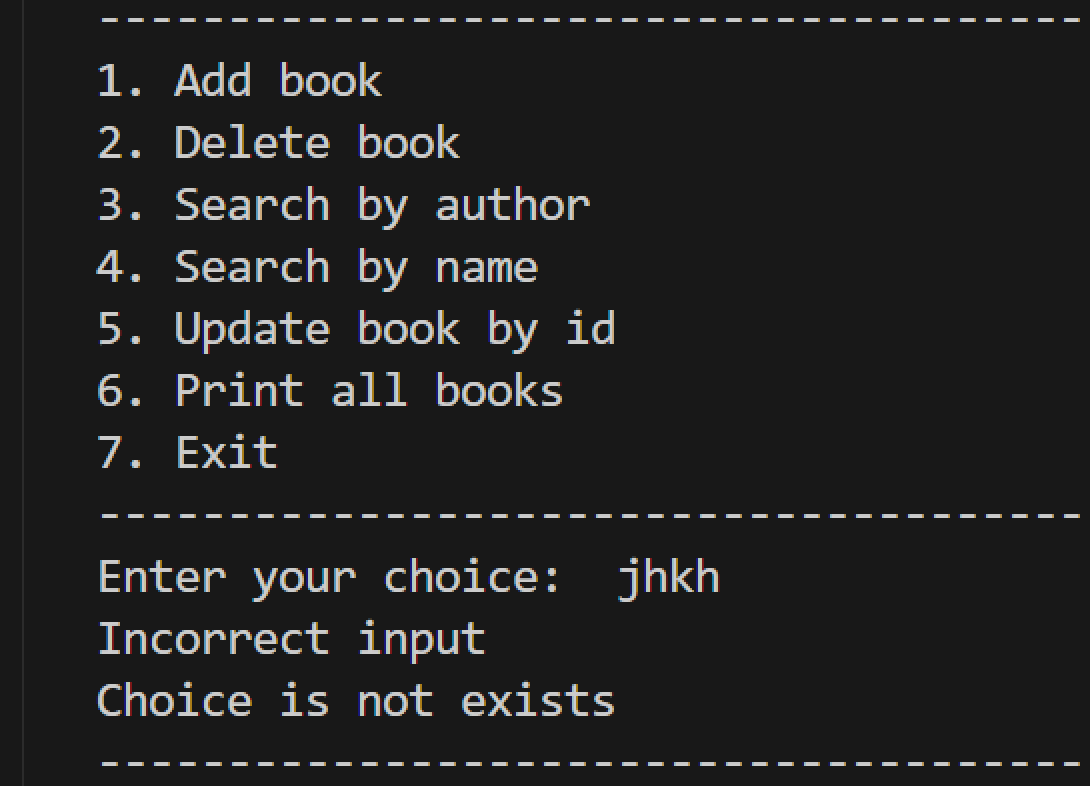
Видно что у этой книги данные изменились

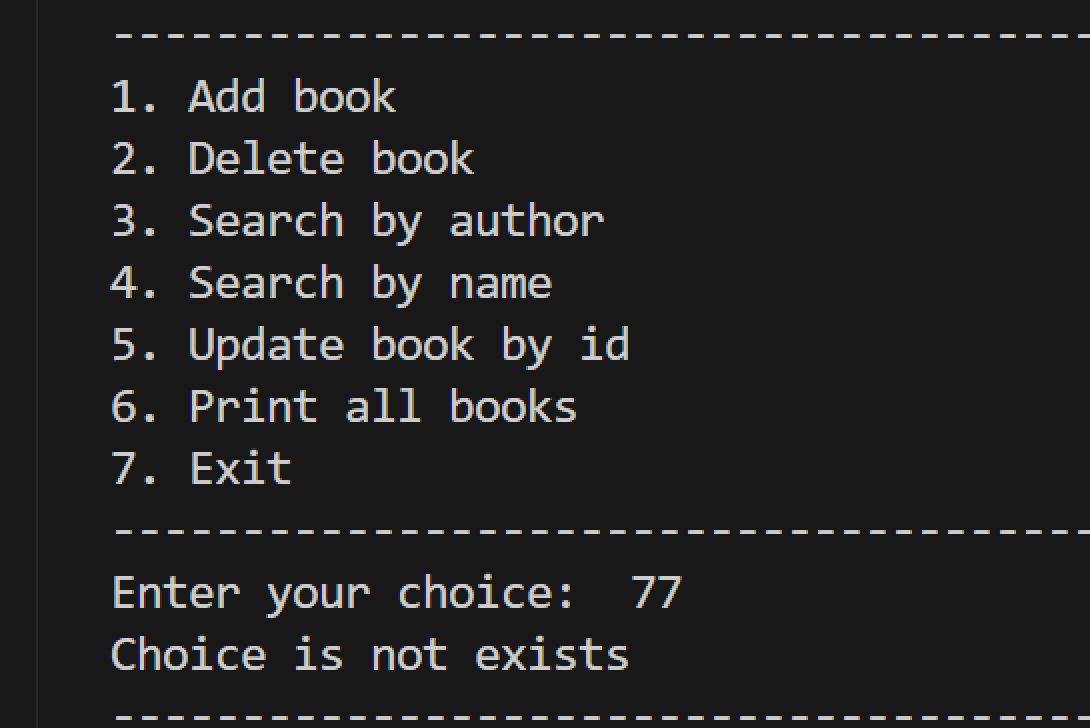


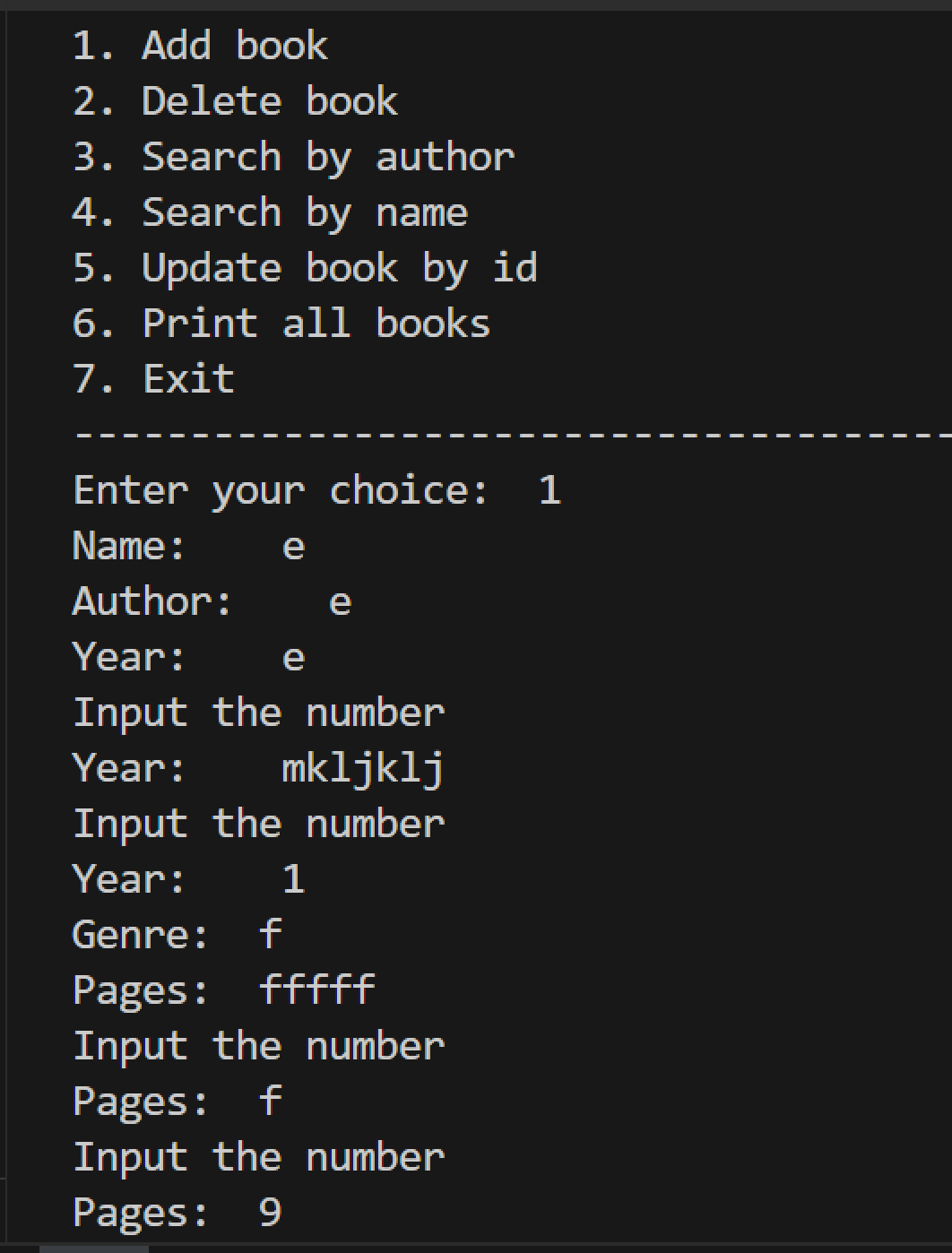
## Выход из программы



## Обработка ошибок







# ВЫВОД

Программа выполняет все требуемые функции, работает с базой данных, корректно соединяется. Ошибки обрабатывает. Нет потери данных и задержек в работе. Приложение удобно для восприятия информации.