**Министерство науки и высшего образования Российской Федерации**

Федеральное государственное автономное образовательное учреждение высшего образования «Уральский федеральный университет имени первого Президента России Б.Н. Ельцина

Институт радиоэлектроники и информационных технологий – РТФ

Центр ускоренного обучения

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №10**

**По дисциплине «Программирование»**

Студент группы РИЗ-140938у Бабенко М.А.

Преподаватель: Архипов Н.А.

**Екатеринбург, 2025 г.**

**Лабораторная работа N10**

**«Введение в алгоритмы и структуры данных. Исследование возможностей**

**коллекций в языке программирования Java»**

Цель: приобретение навыков работы с рекурсивными методами, знакомство с динамическими структурами данных, приобретение навыков создания и использования простейшей динамической структуры, исследование возможностей коллекций в языке программирования Java, сравнение коллекций по времени выполнения операций, знакомство с реализацией коллеккций

**Задания для самостоятельной работы**

**Задание 1.** 1 Сделайте в текстовом редакторе свой файл в формате XML в зависимости от варианта (Приложение 1). 2 Добавьте возможность записывать новые книги в XML-файл. Например, пользователь может ввести данные о новой книге, а программа добавит новый элемент <book> в XML-файл. 3 Добавьте возможность поиска книг по автору или году издания. Например, пользователь может ввести автора или год издания, а программа выведет список книг, удовлетворяющих этому критерию поиска. 4 Реализуйте функцию удаления книги из XML-файла. Например, пользователь может ввести название книги, которую хочет удалить, и программа удалит соответствующий элемент <book> из XML-файла.

Создается новый проект java **LW\_10\_ex1**. В нем создаются несколько .java файлов.

В классе **Main.java** записывается код программы, отвечающий за «связь» других классов. Так, сначала вызывается класс, отвечающий за создание файла со списком книг

В **create\_the\_list.java** создается список книг. В нем мы создаем две книги. Сначала объявляются библиотеки javax.xml и org.w3c, а также java.io. Далее, создается документ при помощи фабрики **DocumentBuilderFactory**, создающей **DocumentBuilder**. Он создает новый документ, который записывается в «переменную» **doc**. Затем, создается корень xml, в котором <library> это корневой элемент, а командой appendChild(root) его добавляют в документ. Создается элемент <book>, задается атрибут id=”1” – это идентификатор книги. Затем, добавляются вложенные элементы: **<title>** - создаёт тег с названием книги, **createTextNode(...)** – текст внутри тега, **appendChild(...)** – добавляет элемент к **<book>**. То же самое делается для author и year. Затем, командой **root.appendChild(book1)** добавляется в **library**. Далее, создается трянсформер – он отвечает за запись XML в файл, задаются настройки отступов и др.. Затем, указывается откуда и куда писать – **DOMSource и StreamResult**. Командой **transformer.transform(source, result);** сохраняется, после чего появляется соответствующее сообщение.

В классе **new\_books.java** реализовано добавление книг. В отличие от прошлого класса, ведется работа уже с существующим файлом: если файл найдет – оно загружается, если нет – создается новый файл с конем **<books>**. Также создается цикл для ввода данных о книге – года, названия и автора; затем программой атоматически генерируется ее ID инкрементрированием длинны списка книг. Также файл сохраняется каждый раз после добавления новой книги, а не один раз в конце

Далее, в классе search.java происходит поиск по содержимому XML – **doc.getElementsByTagName("book")**. В коде перебираются все <book> элементы из XML файла. Далее, пользователем вводится один из тегов, по которому будет совершаться поиск, и происходит сравнение и вывод подходящей книги по **switch**-выражению с помощью **case в match**. Метод **contains** позволяет находить любое совпадение. Далее выводится информация о найденной книге в консоль, или сообщение о том, что книга не найдена. В классе не сохраняется XML-файл.

Далее, создается класс **book\_remove**, он удаляет **<book>** по введенному ID книги (выводится в прошлом классе) и сохраняет изменения в XML. **book.getParentNode().removeChild(book)** удаляет элемент <book> из XML документа, удаление происходит только тогда, когда **book.getAttribute("id")** совпадает с введенным пользователем. Если книга удалена (if (removed)) – файл сохраняется. В противном случае выведется сообщение об ошибке.

Результат работы программы и ее код показан на рисунке 1.

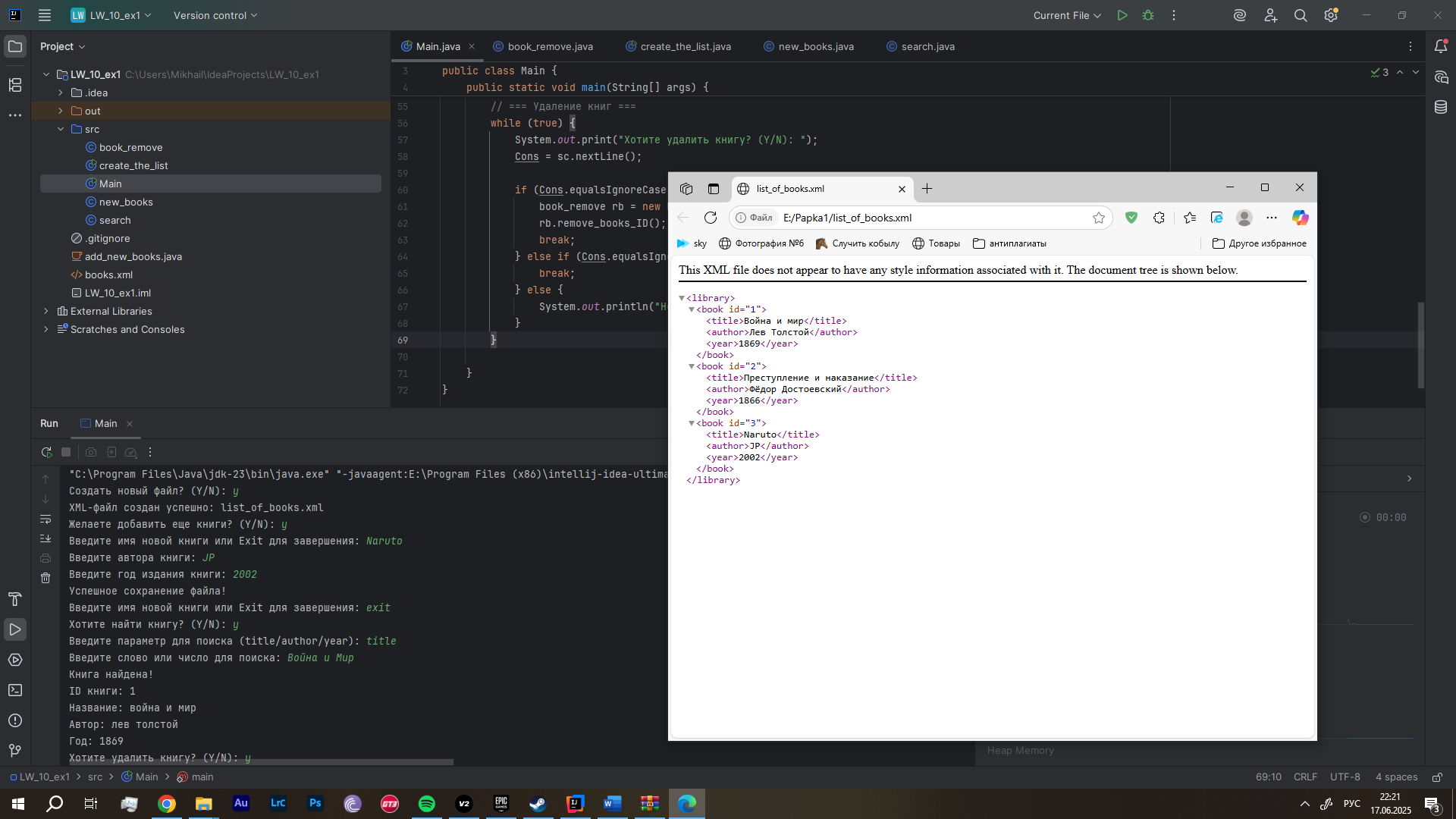


Рисунок 1 – Код и результат работы программы LW\_10\_ex1

Это программа для работы со списком книг. Она сохраняет все данные в файл books.json. При запуске, она сначала спрашивает, нужно ли создать новый файл. Если согласиться, в файл по пути переменной FILE\_PATH запишутся две книжки по умолчанию – метод create() из класса book\_list\_create.

Потом программа предлагает добавить новые книжки через метод addBooks() класса book\_add. Там нужно ввести название, автора и год. Программа сама придумывает каждой книжке ID – количество существующих книг инкрементируется. Всё это сохраняется в тот же файл.

Для поиска книг, работает метод searchBooks() из класса book\_search. Можно искать по названию, автору или году. Для удаления книги, метод removeBookById() в классе book\_remove. Пользователем вводится ID книжки, которую необходимо убрать.

Все эти методы работают с одной и той же переменной FILE\_PATH, где хранится путь к файлу с книжками. Все классы программы связаны классом Main.

Результат работы программы и ее код показан на рисунке 2.

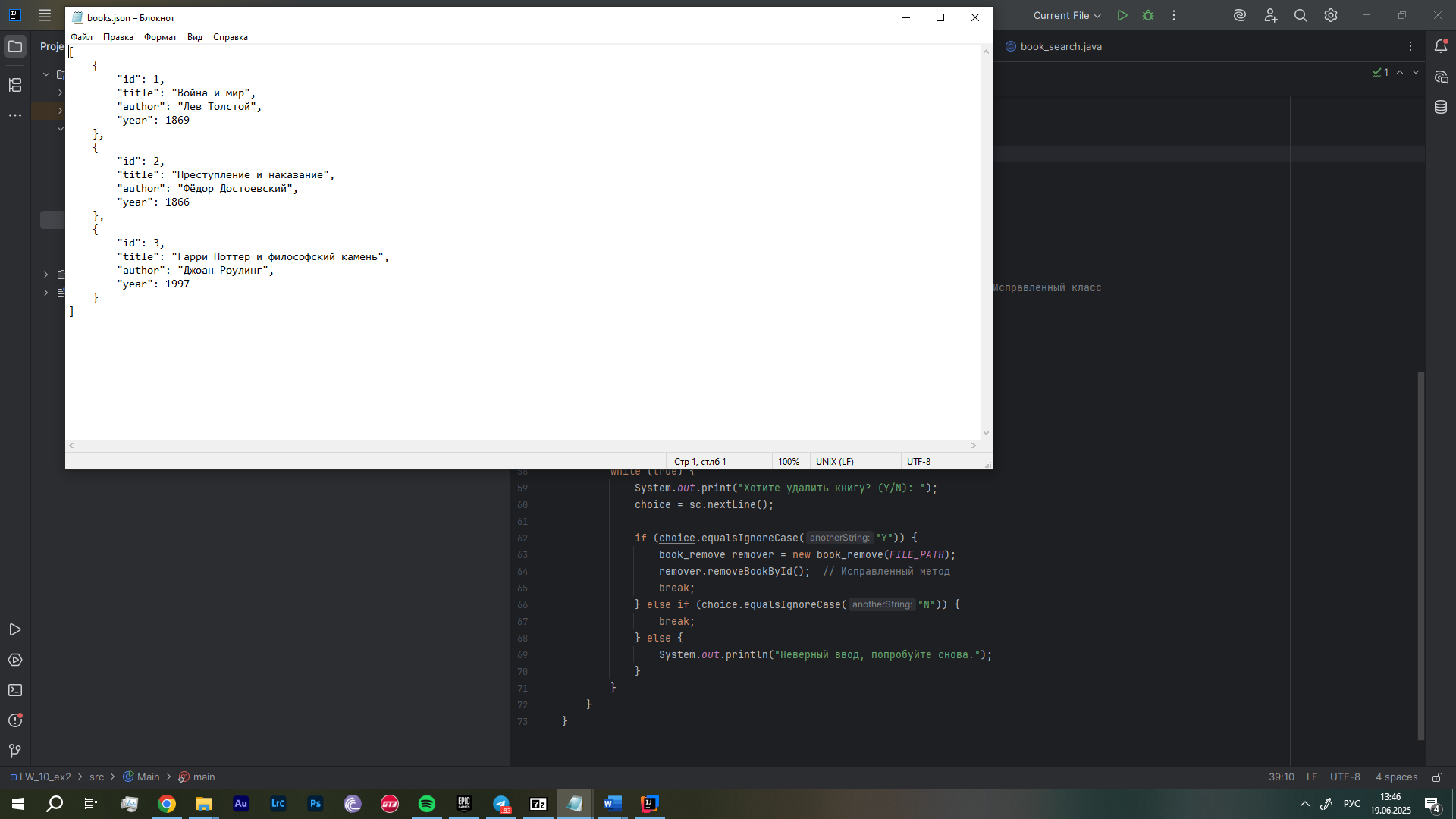


Рисунок 2 – Код и результат работы программы LW\_10\_ex2

LW\_10\_ex3 это список книг, где все данные хранятся в Excel-файле. Состоит из нескольких классов, котоыре вызываются из Main.

Главный класс Main запускает программу и поочерёдно спрашивает пользователя, что он хочет сделать: создать новый файл, добавить книгу, найти книгу или удалить её. Все ответы вводятся через переменную sc типа Scanner, а путь к Excel-файлу сохраняется в переменной FILE\_PATH.

Класс book\_list\_create используется для создания нового Excel-файла, в который сразу записываются две книги. Для работы используются ID, название (title), автор (author) и год издания (year) – они записываются построчно с помощью объектов Workbook, Sheet, Row и Cell.

Класс book\_add добавляет книги в уже существующий Excel-файл. Пользователь по очереди вводит название книги, имя автора и год выпуска, после чего создаётся новая строка в таблице. ID новой книги (nextId) рассчитывается как номер следующей строки, чтобы не было дубликатов.

Класс book\_search отвечает за поиск книги. Пользователь выбирает, по какому параметру искать – названию, автору или году – и вводит значение. Программа перебирает строки Excel-файла, сравнивает значения в нужном столбце и, если находит совпадение, выводит информацию о найденной книге на экран.

Класс book\_remove удаляет книгу по ID. Пользователь вводит нужный ID, программа просматривает строки таблицы, находит строку с таким ID и удаляет её. После этого Excel-файл сохраняется с обновлёнными данными. Во всех классах используется один и тот же путь к файлу, передаваемый через переменную path, и взаимодействие с пользователем происходит через Scanner.

Результат работы программы и ее код показан на рисунке 3

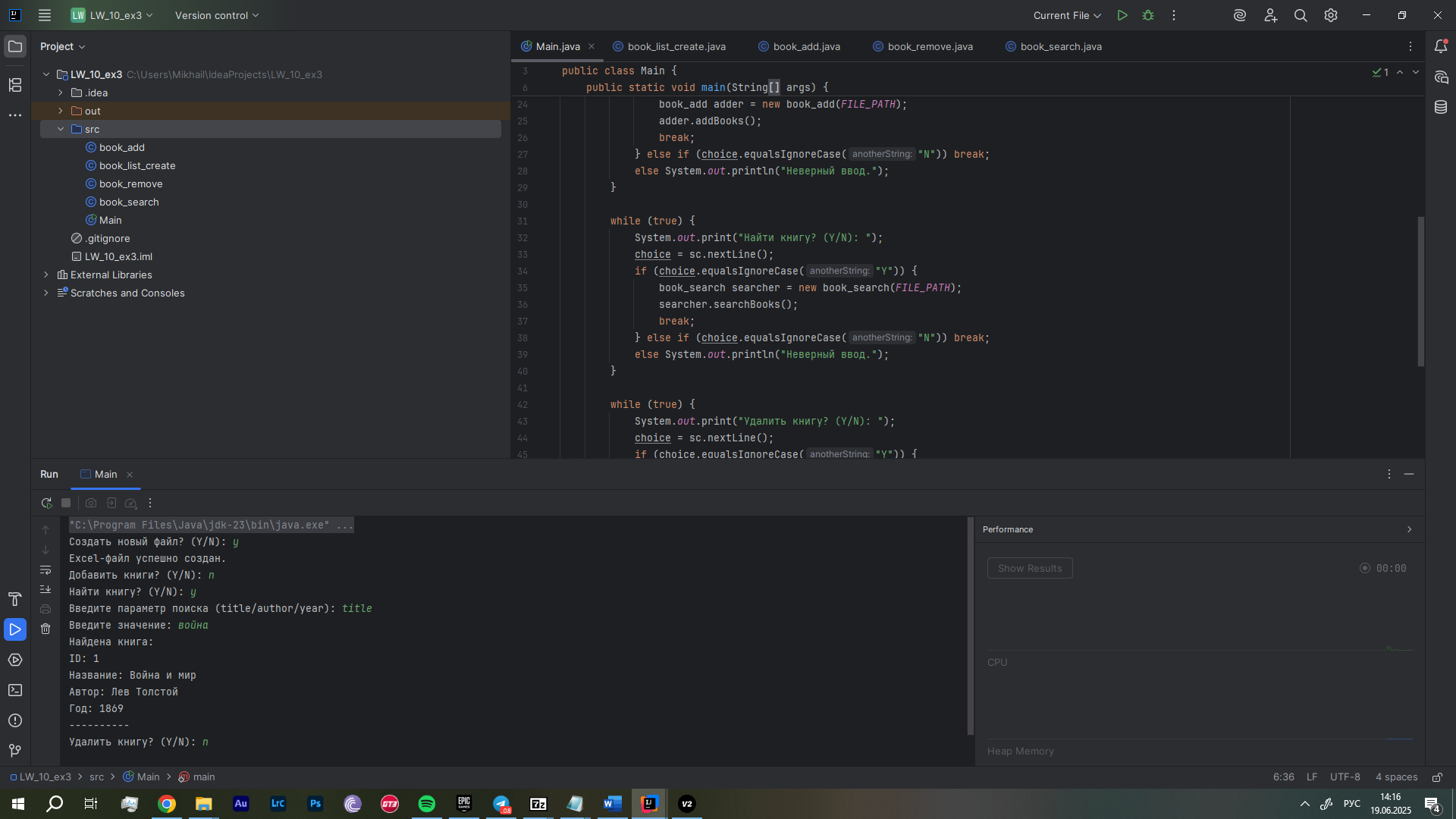


Рисунок 3 – Код и результат работы программы LW\_10\_ex3

**Вывод**

В результате лабораторной работы были приобретены навыки работы с рекурсивными методами, знакомство с динамическими структурами данных, приобретение навыков создания и использования простейшей динамической структуры, исследование возможностей коллекций в языке программирования Java, сравнение коллекций по времени выполнения операций, знакомство с реализацией коллеккций.