**Лабораторная работа «XML»**

1) **Создание XML-документа**

В ходе работы был разработан XML-документ library.xml, содержащий описание библиотеки с тремя книгами. Каждая книга включает такие элементы, как название, автор, год, жанр и цена. Атрибут id сделан уникальным и соответствует требованиям типа xs:ID.

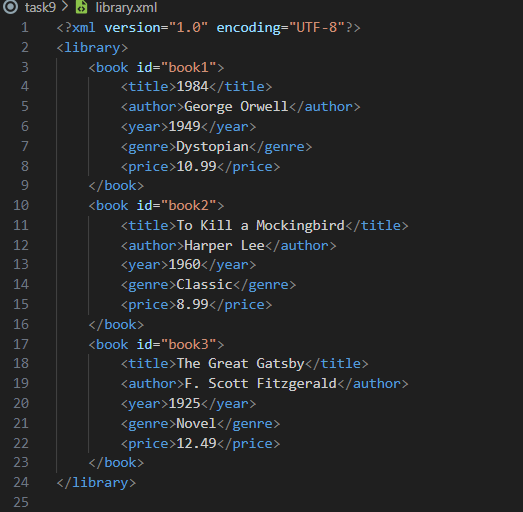


Рисунок 1 – **Код XML-документа library.xml**

2) **Создание XSD-схемы**

Далее была разработана XSD-схема library.xsd для проверки XML-документа. Схема гарантирует:

* Обязательное наличие элементов <title>, <author>, <year>, <genre> и <price>.
* Уникальность и обязательность атрибута id.
* Соответствие значения элемента <year> диапазону от 1800 до текущего года.
* Положительное значение элемента <price> с двумя знаками после запятой.

Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2 **– Код XSD-схемы library.xsd**

3) **Валидация XML-документа**

Следующим этапом стала выполнение проверки XML-документа на соответствие XSD-схеме с использованием Python (библиотека lxml).

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 3 **– Код файла validation.py для проверки XML на соответсвие XSD**

При первой валидации была обнаружена ошибка:



Рисунок 4 **– Ошибка при валидации**

После исправление кода в XML-документе, он стал валиден:



Рисунок 5 – **Результат работы программы после исправления**

4) **Обработка XML-документа**

Далее была написана программа на Python (proc.py) для:

* Чтения и вывода списка всех книг в библиотеке.
* Вычисления средней цены книг.
* Фильтрации книг по заданному критерию (например, жанру или году, в данном примере по жанру «Classic»).

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 6 – **Код программы proc.py**

При выполнении программа показывает следующий результат:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 7 **– Результат работы программы proc.py**

5) **Исследование и сравнение**: Проведен анализ структуры реального XML-документа (sample.xml) на примере RSS-ленты.

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 8 – **Код программы sample.xml**

**Анализ структуры документа**:

1. Корневой элемент <rss> задает формат документа. Атрибут version указывает версию RSS (2.0).
2. Основное содержимое описывается тегом <channel>, который содержит информацию о канале (заголовок, описание, язык).
3. Статьи представляются через повторяющиеся элементы <item>, каждая из которых включает такие теги, как <title>, <link>, <description>, <pubDate> и <guid>.
4. Вложенность логична, что облегчает машинную обработку.
5. Применяются уникальные идентификаторы (<guid>) для каждого <item>, что полезно для управления данными.
6. Цель документа — предоставление пользователям возможности подписываться на обновления (новости) через RSS.

Этот документ четко структурирован и удобен для автоматической обработки, что делает его эффективным форматом для распространения новостей или обновлений.