ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«УФИМСКИЙ УНИВЕРСИТЕТ НАУКИ И ТЕХНОЛОГИЙ»

Институт информатики, математики и робототехники

Кафедра математического и компьютерного моделирования

**Лабораторная работа №6: Работа с форматом XML в Python**

**ОБУЧАЮЩЕГОСЯ**

4 курса группы ПИ-4ИВТ221Б

Санникова Михаила Александровича

|  |  |
| --- | --- |
| Уровень высшего образования:  Направление подготовки (специальность)  Направленность (профиль)  программы  Дата выполнения | высшее образование – бакалавриат  09.03.03 “Прикладная информатика”  Информационные и вычислительные технологии  20.10.2025 |
|  |  |
|  |  |

УФА – 2025

**Постановка задачи:**

Создать файл library.xml

Написать функцию для чтения файла

Добавить в файл library.xml:

• Новую книгу: id="3", title="Машинное обучение", author="Сидорова С.С.", year="2023", genre="Искусственный интеллект"

• Обновите год издания книги с id=2 на 2021

• Добавьте атрибут is\_available="true" для всех книг

Написать функцию, которая проверяет:

• Наличие обязательных элементов у каждой книги (title , author , year, genre)

• Корректность типов данных (year - число, id - число)

• Уникальность идентификаторов книг

Написать функцию, которая реализует поиск книг по различным критериям (по автору, по жанру, по году издания)

**Практическая часть:**

Листинг кода с комментариями

*import* xml.etree.ElementTree *as* ET  
*import* os  
*from* functools *import* wraps  
  
  
*def* main():  
 filename = 'library.xml'  
  
 *# Задание 1: Создание XML-файла*  
create\_initial\_xml(filename)  
  
 *# Задание 2: Чтение XML-файла*  
 *print*("=" \* 60)  
 *print*("ЧТЕНИЕ XML-ФАЙЛА")  
 *print*("=" \* 60)  
 books\_data = read\_xml\_file(filename)  
  
 *# Задание 3: Модификация XML-файла*  
 *print*("\n" + "=" \* 60)  
 *print*("МОДИФИКАЦИЯ XML-ФАЙЛА")  
 *print*("=" \* 60)  
 modified\_data = modify\_xml\_file(filename)  
  
 *# Задание 4: Валидация данных*  
 *print*("\n" + "=" \* 60)  
 *print*("ВАЛИДАЦИЯ ДАННЫХ")  
 *print*("=" \* 60)  
 *if* is\_data\_valid(modified\_data):  
 *print*("✓ Все данные прошли валидацию успешно!")  
 *else*:  
 *print*("✗ Данные содержат ошибки!")  
  
 *# Чтение обновленного файла*  
 *print*("\n" + "=" \* 60)  
 *print*("ОБНОВЛЕННЫЕ ДАННЫЕ")  
 *print*("=" \* 60)  
 updated\_books = read\_xml\_file(filename)  
  
 *# Задание 5: Поиск данных*  
 *print*("\n" + "=" \* 60)  
 *print*("ПОИСК ДАННЫХ")  
 *print*("=" \* 60)  
  
 *# Поиск по автору*  
author\_books = search\_books(updated\_books, author="Петров П.П.")  
 *print*(f"Книги автора 'Петров П.П.': {*len*(author\_books)}")  
 *for* book *in* author\_books:  
 *print*(f" - {book['title']} ({book['year']} год)")  
  
 *# Поиск по жанру*  
genre\_books = search\_books(updated\_books, genre="Программирование")  
 *print*(f"\nКниги жанра 'Программирование': {*len*(genre\_books)}")  
 *for* book *in* genre\_books:  
 *print*(f" - {book['title']} - {book['author']}")  
  
 *# Поиск по году*  
year\_books = search\_books(updated\_books, min\_year=2021)  
 *print*(f"\nКниги изданные с 2021 года: {*len*(year\_books)}")  
 *for* book *in* year\_books:  
 *print*(f" - {book['title']} ({book['year']} год)")  
  
  
*def* create\_initial\_xml(filename):  
 *"""Задание 1: Создание начального XML-файла"""*  
 *# Создание корневого элемента*  
library = ET.Element('library')  
  
 *# Создание первой книги*  
book1 = ET.SubElement(library, 'book')  
 book1.set('id', '1')  
  
 title1 = ET.SubElement(book1, 'title')  
 title1.text = 'Python для начинающих'  
  
 author1 = ET.SubElement(book1, 'author')  
 author1.text = 'Иванов И.И.'  
  
 year1 = ET.SubElement(book1, 'year')  
 year1.text = '2022'  
  
 genre1 = ET.SubElement(book1, 'genre')  
 genre1.text = 'Программирование'  
  
 *# Создание второй книги*  
book2 = ET.SubElement(library, 'book')  
 book2.set('id', '2')  
  
 title2 = ET.SubElement(book2, 'title')  
 title2.text = 'Алгоритмы и структуры данных'  
  
 author2 = ET.SubElement(book2, 'author')  
 author2.text = 'Петров П.П.'  
  
 year2 = ET.SubElement(book2, 'year')  
 year2.text = '2020'  
  
 genre2 = ET.SubElement(book2, 'genre')  
 genre2.text = 'Компьютерные науки'  
  
 *# Создание XML-дерева и запись в файл*  
tree = ET.ElementTree(library)  
 tree.write(filename, encoding='utf-8', xml\_declaration=*True*)  
  
 *print*(f"✓ Создан файл {filename}")  
  
  
*def* check\_file\_exists(func):  
 @wraps(func)  
 *def* wrapper(file\_path):  
 *if not* os.path.exists(file\_path):  
 *print*(f"Не существует файла с заданным путем: {file\_path}")  
 *return None*  
 *return* func(file\_path)  
  
 *return* wrapper  
  
  
@check\_file\_exists  
*def* read\_xml\_file(filename):  
 *"""Задание 2: Чтение XML-файла"""*  
 *try*:  
 tree = ET.parse(filename)  
 root = tree.getroot()  
  
 books = []  
 *print*("Книги в библиотеке:")  
 *print*("-" \* 50)  
  
 *for* book *in* root.findall('book'):  
 book\_id = book.get('id')  
 title = book.find('title').text  
 author = book.find('author').text  
 year = *int*(book.find('year').text)  
 genre = book.find('genre').text  
 is\_available = book.get('is\_available', 'Не указан')  
  
 book\_data = {  
 'id': book\_id,  
 'title': title,  
 'author': author,  
 'year': year,  
 'genre': genre,  
 'is\_available': is\_available  
 }  
 books.append(book\_data)  
  
 *print*(f"ID: {book\_id}, Название: {title}")  
 *print*(f" Автор: {author}, Год: {year}")  
 *print*(f" Жанр: {genre}, Доступна: {is\_available}")  
 *print*("-" \* 30)  
  
 *print*(f"\nОбщее количество книг: {*len*(books)}")  
  
 *if* books:  
 newest\_book = *max*(books, key=*lambda* x: x['year'])  
 *print*(f"Самая новая книга: {newest\_book['title']} ({newest\_book['year']} год)")  
  
 *return* books  
  
 *except Exception as* e:  
 *print*(f"Ошибка при чтении файла: {e}")  
 *return None*  
  
  
*def* modify\_xml\_file(filename):  
 *"""Задание 3: Модификация XML-файла"""*  
 *try*:  
 tree = ET.parse(filename)  
 root = tree.getroot()  
  
 *# Добавление новой книги*  
new\_book = ET.SubElement(root, 'book')  
 new\_book.set('id', '3')  
 new\_book.set('is\_available', 'true')  
  
 title = ET.SubElement(new\_book, 'title')  
 title.text = 'Машинное обучение'  
  
 author = ET.SubElement(new\_book, 'author')  
 author.text = 'Сидорова С.С.'  
  
 year = ET.SubElement(new\_book, 'year')  
 year.text = '2023'  
  
 genre = ET.SubElement(new\_book, 'genre')  
 genre.text = 'Искусственный интеллект'  
  
 *print*("✓ Добавлена новая книга: Машинное обучение")  
  
 *# Обновление года издания книги с id=2*  
 *for* book *in* root.findall('book'):  
 *if* book.get('id') == '2':  
 year\_element = book.find('year')  
 year\_element.text = '2021'  
 *print*("✓ Обновлен год издания книги 'Алгоритмы и структуры данных' на 2021")  
  
 *# Добавление атрибута is\_available="true" для всех книг*  
 *for* book *in* root.findall('book'):  
 *if* book.get('is\_available') *is None*:  
 book.set('is\_available', 'true')  
  
 *print*("✓ Добавлен атрибут is\_available для всех книг")  
  
 *# Запись обновленных данных*  
tree.write(filename, encoding='utf-8', xml\_declaration=*True*)  
 *print*("✓ Файл успешно обновлен!")  
  
 *# Возвращаем обновленные данные*  
 *return* read\_xml\_file(filename)  
  
 *except Exception as* e:  
 *print*(f"Ошибка при модификации файла: {e}")  
 *return None*  
  
  
*def* is\_data\_valid(books):  
 *"""Задание 4: Валидация данных"""*  
 *if not* books:  
 *return False*  
  
errors = []  
 book\_ids = *set*()  
  
 *for* book *in* books:  
 *# Проверка обязательных элементов*  
required\_fields = ['title', 'author', 'year', 'genre']  
 *for* field *in* required\_fields:  
 *if* field *not in* book *or not* book[field]:  
 errors.append(f"Книга ID {book['id']}: отсутствует обязательное поле '{field}'")  
  
 *# Проверка уникальности идентификаторов*  
 *if* book['id'] *in* book\_ids:  
 errors.append(f"Дублирующийся ID книги: {book['id']}")  
 book\_ids.add(book['id'])  
  
 *# Проверка корректности типов данных*  
 *if not* book['id'].isdigit():  
 errors.append(f"Книга '{book['title']}': ID должен быть числом")  
  
 *if not isinstance*(book['year'], *int*) *or* book['year'] <= 0:  
 errors.append(f"Книга '{book['title']}': год должен быть положительным целым числом")  
  
 *# Проверка атрибута is\_available*  
 *if* book.get('is\_available') *not in* ['true', 'false']:  
 errors.append(f"Книга '{book['title']}': неверное значение is\_available")  
  
 *# Вывод результатов валидации*  
 *if* errors:  
 *print*("Найдены ошибки валидации:")  
 *for* error *in* errors:  
 *print*(f" ✗ {error}")  
 *return False*  
 *else*:  
 *return True*  
  
  
*def* search\_books(books, author=*None*, genre=*None*, min\_year=*None*, max\_year=*None*):  
 *"""Задание 5: Поиск данных по различным критериям"""*  
filtered\_books = []  
  
 *for* book *in* books:  
 *# Проверка критерия автора*  
 *if* author *and* book['author'] != author:  
 *continue*  
  
 *# Проверка критерия жанра*  
 *if* genre *and* book['genre'] != genre:  
 *continue*  
  
 *# Проверка критерия года*  
 *if* min\_year *and* book['year'] < min\_year:  
 *continue*  
  
 *if* max\_year *and* book['year'] > max\_year:  
 *continue*  
  
filtered\_books.append(book)  
  
 *return* filtered\_books  
  
  
*if* \_\_name\_\_ == '\_\_main\_\_':  
 main()

Скриншоты выполнения программы

