МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ЗВІТ ПРО ВИКОНАННЯ

КОНТРОЛЬНОЇ РОБОТИ

із дисципліни: «Програмування мовою Python»

на тему «Парсинг XML-файлів»

Виконав студент групи КН-32 Ярош Ярослав Олександрович

Варіант 7

Перевірила Парфененко Ю.В.

Суми 2024

**ЗМІСТ**

[**ВСТУП** 3](#_Toc183427037)

[**1** **АНАЛІЗ ВИКОРИСТАННЯ БІБЛІОТЕК XML, PANDAS, MATPLOTLIB ТА COUNTER ДЛЯ ВИРІШЕННЯ ЗАДАЧІ ПАРСИНГУ XML-ФАЙЛІВ** 4](#_Toc183427038)

[1.1 Модулі xml, pandas, matplotlib та counter 4](#_Toc183427039)

[**2** **РОЗВ’ЯЗАННЯ ЗАДАЧІ ПАРСИНГУ XML-ФАЙЛІВ** 5](#_Toc183427040)

[2.1 Постановка задачі 5](#_Toc183427041)

[2.2 Алгоритм розв’язання задачі 6](#_Toc183427042)

[2.3 Програмна реалізація 6](#_Toc183427043)

[**ВИСНОВКИ** 14](#_Toc183427044)

[**СПИСОК ЛІТЕРАТУРИ** 15](#_Toc183427045)

# **ВСТУП**

Мова програмування Python – високорівнева мова програмування загального призначення, яка орієнтована на простоту коду та продуктивність розробника.

Обробка даних є важливою складовою сучасного програмування, зокрема для аналізу та візуалізації інформації. У цьому індивідуальному домашньому завданні розроблено програму для роботи з даними про книги, представленими у форматі XML. Програма виконує парсинг файлу, відображає дані у табличному форматі та візуалізує їх за роками публікації й жанрами з побудовою графіків.

# **АНАЛІЗ ВИКОРИСТАННЯ БІБЛІОТЕК XML, PANDAS, MATPLOTLIB ТА COUNTER ДЛЯ ВИРІШЕННЯ ЗАДАЧІ ПАРСИНГУ XML-ФАЙЛІВ**

## Модулі xml, pandas, matplotlib та counter

Розв’язання задач обробки даних часто потребує використання спеціалізованих бібліотек, які забезпечують зручність і ефективність роботи. У цьому завданні для парсингу XML-файлів, аналізу даних і візуалізації використано бібліотеки xml, pandas, matplotlib та модуль collections.Counter. Кожна з цих бібліотек виконує свою функцію

Таблиця 1.1 - Порівняльна таблиця модулів

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Критерій** | **xml** | **pandas** | **matplotlib** | **collections.Counter** |
| **Призначення** | Робота з XML-файлами | Вивід табличних даних | Візуалізація даних | Підрахунок частоти елементів |
| **Складність використання** | Низька | Середня | Середня | Низька |
| **Продуктивність** | Середня | Висока | Висока | Висока |
| **Особливі функції** | Робота з деревоподібними структурами | Робота з DataFrame | Побудова графіків | Простий підрахунок |

# **РОЗВ’ЯЗАННЯ ЗАДАЧІ ПАРСИНГУ XML-ФАЙЛІВ**

## 2.1 Постановка задачі

Необхідно створити програму для обробки даних, представлених у форматі XML. Програма повинна зчитувати XML-файл, витягувати ключову інформацію даних, групувати дані за заданими критеріями та відображати результати у вигляді таблиць і графіків.

Користувач зможе переглядати всі дані, графіки розподілу даних і взаємодіяти з програмою через меню. Завдання включає використання Python-бібліотек для парсингу XML, аналізу та візуалізації даних, що дозволить вирішити поставлені задачі.

Для прикладу виконання задачі було обрано прикладовий файл з даними у форматі XML, розміром 1МБ з веб сайту <https://examplefile.com/code/xml/1-mb-xml>.

Рисунок 2.1 - Приклад вигляду сайту, з якого взято файл XML

A screenshot of a computer

Description automatically generated

## 2.2 Алгоритм розв’язання задачі

1. Імпортувати бібліотеки.

2. Написати функцію для зчитування XML-файлу (parse\_file).

3. Розробити функцію для відображення книг у вигляді таблиці (show\_all\_books).

4. Реалізувати функції для побудови графіків (show\_plot\_by\_year, show\_plot\_by\_genre).

5. Створити функцію меню для взаємодії з користувачем (menu).

## 2.3 Програмна реалізація

Таблиця 2.1 - Таблиця компонентів програми

|  |  |  |  |
| --- | --- | --- | --- |
| **Назва** | **Тип** | **Опис** | **Призначення** |
| xml.etree.ElementTree | Бібліотека | Бібліотека для роботи з XML-файлами | Парсинг XML-файлу та отримання необхідних даних |
| pandas | Бібліотека | Бібліотека для роботи з табличними даними | Формування та відображення даних у вигляді таблиць |
| matplotlib.pyplot | Бібліотека | Бібліотека для побудови графіків і візуалізації даних | Створення стовпчикових і кругових діаграм |
| collections.Counter | Модуль | Модуль для підрахунку частоти елементів у послідовностях | Підрахунок кількості елементів для розподілу книг за роками та жанрами |
| parse\_file | Метод | Функція для парсингу XML-файлу та збереження даних у вигляді списку словників | Отримання з XML-файлу даних про книги, такі як назва, автор, жанр та рік публікації |
| show\_all\_books | Метод | Функція для виведення всіх книг у табличному форматі | Відображає список книг у вигляді таблиці за допомогою pandas |
| show\_plot\_by\_year | Метод | Функція для створення стовпчикового графіка розподілу книг за роками | Аналіз даних про книги за роками та візуалізація їх за допомогою matplotlib |
| show\_plot\_by\_genre | Метод | Функція для створення кругової діаграми розподілу книг за жанрами | Відображення співвідношення книг за жанрами у вигляді кругової діаграми |
| menu | Метод | Функція для взаємодії користувача з програмою через текстове меню | Забезпечення вибору дії: перегляд книг, створення графіків чи вихід з програми |

Таблиця 2.2 – Таблиця символьних імен

|  |  |  |
| --- | --- | --- |
| **Назва** | **Тип даних** | **Призначення** |
| filename | str | Ім’я XML-файлу |
| tree, root | Object | Дерево та корінь XML-структури |
| books | list | Список з даними про книги |
| df | DataFrame | Табличне представлення книг |
| year\_counts, genre\_counts | Counter | Частотний розподіл книг за роками та жанрами |

**Текст програми**

# Імпортування необхідних бібліотек

import xml.etree.ElementTree as ET

from fileinput import filename

import pandas as pd

import matplotlib.pyplot as plot

from collections import Counter

# Функція для парсингу XML-файлу

def parse\_file(filename):

# Завантаження XML-файлу та отримання його кореневого елемента

tree = ET.parse(filename)

root = tree.getroot()

# Ініціалізація списку для зберігання інформації про книги

books = []

# Ітерація по кожному елементу <book> у файлі

for book in root.findall('book'):

# Збереження даних про книгу у вигляді словника

books.append({

"Title": book.find("title").text, # Назва книги

"Author": book.find("author").text, # Автор книги

"Genre": book.find("genre").text, # Жанр книги

"Year": book.find("year").text # Рік публікації

})

# Повернення списку книг

return books

# Функція для відображення всіх книг у табличному форматі

def show\_all\_books(books):

# Перетворення списку словників у DataFrame (таблицю) та її відображення

df = pd.DataFrame(books)

print(df)

# Функція для побудови графіка розподілу книг за роками

def show\_plot\_by\_year(books):

# Підрахунок кількості книг для кожного року за допомогою Counter

year\_counts = Counter(book["Year"] for book in books)

# Сортування років для коректного відображення на графіку

years = sorted(year\_counts.keys())

counts = [year\_counts[year] for year in years]

# Створення стовпчикового графіка

plot.figure(figsize=(10, 6)) # розмір графіка

plot.bar(years, counts) # побудова стовпчиків

plot.title("Розподіл книг за роками") # заголовок графіка

plot.xlabel("Рік") # підпис осі X

plot.ylabel("Кількість книг") # підпис осі Y

plot.xticks(rotation=45) # нахил підписів на осі X для читабельності

plot.grid(axis='y', linestyle="solid", alpha=0.5) # додавання сітки для осі Y

plot.show() # відображення графіка

# Функція для побудови кругової діаграми розподілу книг за жанрами

def show\_plot\_by\_genre(books):

# Підрахунок кількості книг для кожного жанру за допомогою Counter

genre\_counts = Counter(book["Genre"] for book in books)

# Отримання списків жанрів і їх кількостей

labels = list(genre\_counts.keys())

sizes = list(genre\_counts.values())

# Створення кругової діаграми

plot.figure(figsize=(8, 8))

plot.pie(sizes, labels=labels)

plot.title("Розподіл книг за жанрами")

plot.show()

# Функція для відображення меню користувача та взаємодії з програмою

def menu(books):

while True:

# Виведення меню з доступними опціями

print("\nМеню:")

print("1. Вивести всі книги")

print("2. Показати графік розподілу за роками")

print("3. Показати графік розподілу за жанрами")

print("4. Вийти")

choice = input("Оберіть опцію (1-4): ") # Запит вибору у користувача

# Обробка вибору користувача

if choice == "1":

show\_all\_books(books)

elif choice == "2":

show\_plot\_by\_year(books)

elif choice == "3":

show\_plot\_by\_genre(books)

elif choice == "4":

break

else:

print("Некоректний вибір") # Повідомлення про некоректний ввід

# Основна частина програми

filename = "data.xml" # Змінна з назвою файла

books = parse\_file(filename) # Виклик функції парсингу для зчитування даних із XML-файлу

menu(books) # Запуск меню для взаємодії з користувачем

**Контрольний приклад**

A screenshot of a computer program

Description automatically generated

Рисунок 2.2 – XML-файл

A screen shot of a computer

Description automatically generated

Рисунок 2.3 – Меню для взаємодії з користувачем

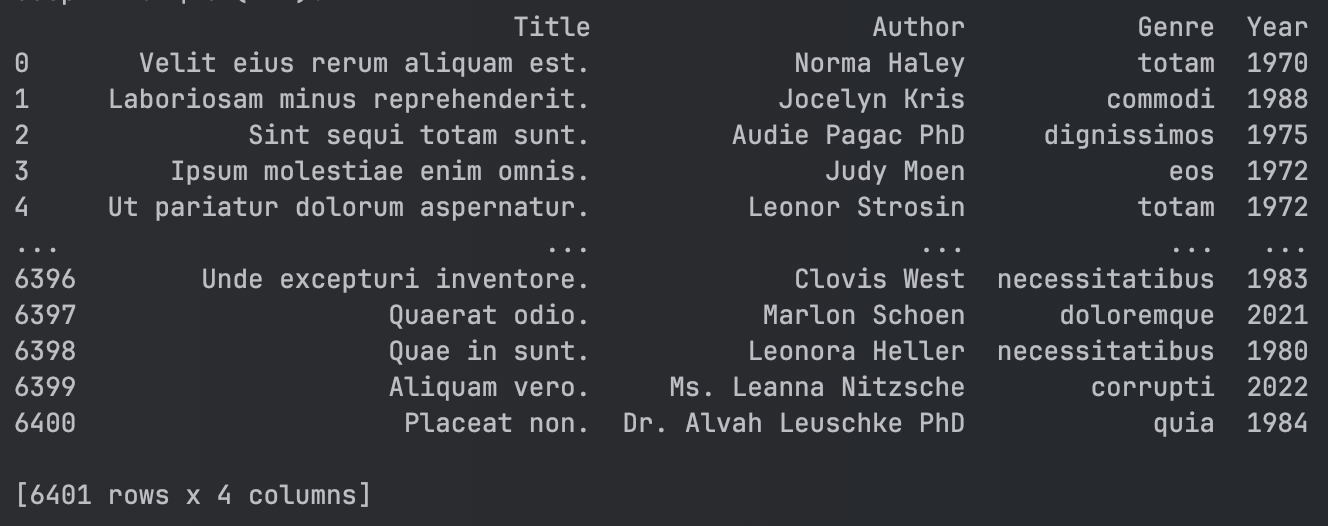


Рисунок 2.3 – Вивід книг в табличному вигляді

A graph of blue bars and numbers

Description automatically generated

Рисунок 2.3 – Стовпчастий графік розподілу книг за роками

A colorful circle with text around it

Description automatically generated

Рисунок 2.3 – Круговий графік розподілу книг за жанрами

# **ВИСНОВКИ**

У результаті виконання індивідуального домашнього завдання була реалізована програма для парсингу XML-файлів. Було використано бібліотеки Python для обробки XML-файлів, аналізу даних і візуалізації результатів. Програма, що була розроблена, здійснює парсинг даних, представляє їх у табличному вигляді та будує графіки розподілу книг за роками та жанрами. Для цього застосовувалися бібліотеки xml.etree.ElementTree для парсингу XML, pandas для створення таблиць, matplotlib для візуалізації, а також collections.Counter для підрахунку частоти елементів.

Це завдання сприяло не лише освоєнню нових інструментів для роботи з даними, але й розвитку навичок програмування, що можуть бути застосовані в практиці, в різних інших задачах.

# **СПИСОК ЛІТЕРАТУРИ**

1. XML Example File – <https://examplefile.com/code/xml/1-mb-xml>
2. XML Parsing – <https://www.geeksforgeeks.org/xml-parsing-python/>