Практическое занятие № 4. Что вы можете сказать об этих фильмах?

В этом задании вы будете практиковаться в чтении и обработке xml-файлов. Для этого в Python существует библиотека `lxml`. Вы можете ее подключить при помощи команды

```
from lxml import etree
```

1. Исходные данные

Вам будет предложено два xml-файла, которые представляют выгрузку информации о фильмах из базы данных IMDB. Первый файл содержит небольшой фрагмент выгрузки и его лучше использовать в процессе разработки программы. Этот набор данных можно скачать по адресу http://bit.ly/2dgWxbg. Когда ваша программа будет отлажена, используйте ее для анализа полного набора данных, который можно скачать по адресу http://bit.ly/2cAhv9p.

Предлагаемые вам файлы имеют следующую структуру (чтобы в этом убедиться, можете открыть их в самом простом текстовом редакторе):

```
<?xml version='1.0' encoding='utf-8'?>
<collection>
  <movie> <!--- Описание одного фильма --->
    <title>'Breaker' Morant </title> <!--- Название фильма --->
    <year>1980</year> <!--- Год выпуска --->
    <cast num="42"> <!--- Список актеров, num - сколько актеров снималось в фильме --->
      <actor>Brown, Bryan (I)</actor> <!--- Имя актера --->
      <actor>Henderson, Dick (II)</actor>
      <actor>Gray, Ian (I)</actor>
      ...
    </cast>
  </movie>
  <movie> <!--- Следующий фильм --->
    ...
  </movie>
  ...
</collection>
```

2. Задача

Напишите программу, которая позволит вам ответить на следующие вопросы:

- Найдите три года, в которые выпускалось наибольшее/наименьшее количество картин?
- Какой актер играл в наибольшем количестве фильмов?
- У какого актера самая длинная кинематографическая карьера?

3. Полезные факты о модуле `lxml`

1. Загрузить файл и построить DOM-дерево можно при помощи функции `etree.parse(filename)`, где `filename` — имя xml-файла.
2. Как только вы построили DOM-дерево `doc`, получить корневой элемент можно при помощи метода `doc.getroot()`.
3. У элемента дерева `node` есть два полезных атрибута: `node.tag` (xml-тег элемента) и `node.text` (текст, который находится между открывающим и закрывающим тегом, включая пробелы и переводы строки).
4. Рекурсивно обойти дерево начиная с узла `node` можно при помощи такой конструкции:

   ```
   for item in node.iter():
       # обрабатываем узел item
   ```

5. Больше примеров есть в Google.