Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

*Факультет программной инженерии и компьютерной техники*

**Лабораторная работа №4**

Исследование протоколов, форматов обмена информацией и языков разметки документов

Вариант №12

Группа: P3132

Выполнил:

Волков Григорий Алексеевич

Проверил:

к.т.н. преподаватель

Белозубов Александр Владимирович

г. Санкт-Петербург

2022г.

Оглавление

[Оглавление 2](#_Toc121301795)

[Задание 3](#_Toc121301796)

[Порядок выполнения работы 4](#_Toc121301797)

[Обязательное задание 6](#_Toc121301798)

[Дополнительное задание №1 8](#_Toc121301799)

[Дополнительное задание №2 10](#_Toc121301800)

[Дополнительное задание №3 12](#_Toc121301801)

[Заключение 13](#_Toc121301802)

Задание

|  |  |  |  |
| --- | --- | --- | --- |
| № варианта | Исходный формат | Результирующий формат | День недели |
| 12 | JSON | XML | Среда |

Порядок выполнения работы

Определить номер варианта как остаток деления на 36 порядкового номера в списке группы в ISU. В случае, если в данный день недели нет занятий, то увеличить номер варианта на восемь.

Изучить форму Бэкуса-Наура.

Изучить особенности языков разметки/ форматов JSON, YAML, XML.

Понять устройство страницы с расписанием для своей группы:

http://itmo.ru/ru/schedule/0/P3110/schedule.htm

Исходя из структуры расписания конкретного дня, сформировать файл с расписанием в формате, указанном в задании в качестве исходного. При этом необходимо, чтобы в выбранном дне было не менее двух занятий (можно использовать своё персональное). В случае, если в данный день недели нет таких занятий, то увеличить номер варианта ещё на восемь.

Обязательное задание (позволяет набрать до 65 процентов от максимального числа баллов БаРС за данную лабораторную): написать программу на языке Python 3.x, которая бы осуществляла парсинг и конвертацию исходного файла в новый.

Нельзя использовать готовые библиотеки, в том числе регулярные выражения в Python и библиотеки для загрузки XML-файлов.

Дополнительное задание №1 (позволяет набрать +10 процентов от максимального числа баллов БаРС за данную лабораторную).

a)  Найти готовые библиотеки, осуществляющие аналогичный парсинг и конвертацию файлов.

b)  Переписать исходный код, применив найденные библиотеки. Регулярные выражения также нельзя использовать.

c)  Сравнить полученные результаты и объяснить их сходство/различие.

Дополнительное задание №2 (позволяет набрать +10 процентов от максимального числа баллов БаРС за данную лабораторную).

a) Переписать исходный код, добавив в него использование регулярных выражений.

b) Сравнить полученные результаты и объяснить их сходство/различие.

Дополнительное задание №3 (позволяет набрать +10 процентов от максимального числа баллов БаРС за данную лабораторную).

a)  Используя свою исходную программу из обязательного задания, программу из дополнительного задания №1 и программу из дополнительного задания №2, сравнить стократное время выполнения парсинга + конвертации в цикле.

b)  Проанализировать полученные результаты и объяснить их сходство/различие.

11.Дополнительное задание №4 (позволяет набрать +5 процентов от максимального числа баллов БаРС за данную лабораторную).

c) Переписать исходную программу, чтобы она осуществляла парсинг и конвертацию исходного файла в любой другой формат (кроме JSON, YAML, XML, HTML): PROTOBUF, TSV, CSV, WML и т.п.

d) Проанализировать полученные результаты, объяснить особенности использования формата.

Обязательное задание

Написать программу на языке Python 3.x, которая бы осуществляла парсинг и конвертацию JSON файла в XML.

Исходный код

import time  
  
starttime = time.perf\_counter() # Запускаем таймер  
json\_file = open('timetable.json', 'r') # Открываем json-файл для чтения  
xml\_file = open('timetable\_no\_lib.xml', 'w') # Открываем XML-файл для записи  
xml\_file.write('<?xml version="1.0" encoding="UTF-8" ?>\n<root>\n')  
json\_text = json\_file.read()  
json\_text = json\_text.replace('\n', '')  
json\_text = json\_text.replace(' ', ' ') #приводим к виду в одну строку  
quotesnumber = 0 # счетчик кавычек  
iskey = 0  
isznach = 0  
isznachmassive = 0  
keysmassive = ['root']  
key = ''  
znach = ''  
znachmassive = []  
tabcount = 0  
for i in json\_text:  
 if i == '{':  
 tabcount += 1  
 elif i == '}':  
 tabcount -= 1  
 xml\_file.write('\t' \* tabcount + '</' + keysmassive[-1] + '>\n')  
 keysmassive = keysmassive[:-1]  
  
 if i == '"':  
 quotesnumber += 1  
 if quotesnumber == 1:  
 iskey = 1  
 elif quotesnumber == 2:  
 if iskey == 1:  
 keysmassive.append(key.replace(' ', ''))  
 xml\_file.write('\t' \* tabcount + '<' + keysmassive[-1] + '>')  
 iskey = 0  
 key = ''  
 elif quotesnumber == 3:  
 isznach = 1  
 elif quotesnumber == 4:  
 if isznach == 1:  
 xml\_file.write(znach + '</' + keysmassive[-1] + '>\n')  
 keysmassive = keysmassive[:-1]  
 isznach = 0  
 quotesnumber = 0  
 znach = ''  
  
 elif i == '[':  
 isznachmassive = 1  
 isznach = 0  
 elif isznachmassive == 1:  
 if i == '{':  
 xml\_file.write('\n')  
 isznachmassive = 0  
 znachmassive = []  
 znach = ''  
 quotesnumber = 0  
 elif i == ',':  
 znachmassive.append(znach.strip())  
 znach = ''  
 elif i == ']':  
 znachmassive.append(znach.strip())  
 xml\_file.write(znachmassive[0] + '</' + keysmassive[-1] + '>\n')  
 znachmassive = znachmassive[1:]  
 for a in znachmassive:  
 xml\_file.write('\t' \* tabcount + '<' + keysmassive[-1] + '>' + a + '</' + keysmassive[-1] + '>\n')  
 keysmassive = keysmassive[:-1]  
 znach = ''  
 znachmassive = []  
 isznachmassive = 0  
 quotesnumber = 0  
 else:  
 znach += i  
 elif isznach == 1:  
 znach += i  
 elif iskey == 1:  
 key += i  
print(time.perf\_counter() - starttime)  
json\_file.close()  
xml\_file.close()

Результат выполнения программы

<?xml version="1.0" encoding="UTF-8" ?>  
<root>  
 <day>Среда</day>  
 <classes>  
 <firstclass>  
 <time>8:20 - 9:50</time>  
 <week>3</week>  
 <week>7</week>  
 <week>9</week>  
 <week>11</week>  
 <week>13</week>  
 <week>15</week>  
 <week>17</week>  
 <building>Кронверкский пр., д.49, лит.А</building>  
 <room>2306/2 (БЫВШ. 305) АУД.</room>  
 <lesson>ИНФОРМАТИКА(ЛАБ)</lesson>  
 <teacher>Белозубов Александр Владимирович</teacher>  
 <lesson-format>Очно - дистанционный</lesson-format>  
 </firstclass>  
 <secondclass>  
 <time>10:00-11:30</time>  
 <week>3</week>  
 <week>7</week>  
 <week>9</week>  
 <week>11</week>  
 <week>13</week>  
 <week>15</week>  
 <week>17</week>  
 <building>Кронверкский пр., д.49, лит.А</building>  
 <room>2306/2 (БЫВШ. 305) АУД.</room>  
 <lesson>ИНФОРМАТИКА(ЛАБ)</lesson>  
 <teacher>Белозубов Александр Владимирович</teacher>  
 <lesson-format>Очно - дистанционный</lesson-format>  
 </secondclass>  
 </classes>  
</root>

Дополнительное задание №1

Найти готовые библиотеки осуществляющие парсинг и конвертацию файлов, переписать исходный код используя данные библиотеки (без регулярных выражений). Сравнить полученные результаты и объяснить их сходства/различия.

Исходный код

import json # импортируем библотеку json  
import time  
from json2xml import json2xml #подключаем биболтеку для конвертации json-файла в xml-файл  
starttime = time.perf\_counter() #Запускаем таймер  
xml\_file = open(r'timetable\_with\_lib.xml', 'w') #открываем xml-файл для записи  
json\_file = open('timetable.json') # открываем json-файл и называем его как json\_file  
data = json.load(json\_file) #используем метод load, который считывает json-файл и возвращает объект питона(словарь)  
xml\_file.write(json2xml.Json2xml(data, wrapper='root', pretty=True, attr\_type=False, item\_wrap=False).to\_xml()) #используем функцию json2xml для парсинга файла  
xml\_file.close()  
json\_file.close() #закрываем файлы  
print(time.perf\_counter()-starttime) #Выводим время выполнения программы

Результат выполнения программы

<?xml version="1.0" encoding="UTF-8" ?>  
<root>  
 <day>Среда</day>  
 <classes>  
 <firstclass>  
 <time>8:20 - 9:50</time>  
 <week>3</week>  
 <week>7</week>  
 <week>9</week>  
 <week>11</week>  
 <week>13</week>  
 <week>15</week>  
 <week>17</week>  
 <building>Кронверкский пр., д.49, лит.А</building>  
 <room>2306/2 (БЫВШ. 305) АУД.</room>  
 <lesson>ИНФОРМАТИКА(ЛАБ)</lesson>  
 <teacher>Белозубов Александр Владимирович</teacher>  
 <lesson-format>Очно - дистанционный</lesson-format>  
 </firstclass>  
 <secondclass>  
 <time>10:00-11:30</time>  
 <week>3</week>  
 <week>7</week>  
 <week>9</week>  
 <week>11</week>  
 <week>13</week>  
 <week>15</week>  
 <week>17</week>  
 <building>Кронверкский пр., д.49, лит.А</building>  
 <room>2306/2 (БЫВШ. 305) АУД.</room>  
 <lesson>ИНФОРМАТИКА(ЛАБ)</lesson>  
 <teacher>Белозубов Александр Владимирович</teacher>  
 <lesson-format>Очно - дистанционный</lesson-format>  
 </secondclass>  
 </classes>  
</root>

Код с использованием библиотек имеет улучшенную читаемость, чем код написанный без них. Уменьшается количество строк исходного кода при использовании библиотек, они позволяют пропускать уже реализованные в них шаги, как например чтения файла.

Дополнительное задание №2

Переписать исходный код, добавив в него использование регулярных выражений. Сравнить полученные результаты и объяснить их сходства/различия.

Исходный код

import json  
import re  
import time  
starttime = time.perf\_counter() #Запускаем таймер  
json\_file = open('timetable.json', 'r') #Открываем json-файл для чтения  
xml\_file = open('timetable\_regular.xml', 'w') #Открываем XML-файл для записи  
xml\_file.write('<?xml version="1.0" encoding="UTF-8" ?>\n<root>\n')  
json\_text = json\_file.read()  
json\_text = re.sub(r'\n\s\*', ' ', json\_text)  
print(json\_text)  
tabcount = 1  
keymassive = ['root']  
massive = []  
key = ''  
key\_znach\_pattern = re.compile(r'(?:(?<=\")\s\*[^"]\*\":\s\*(?: \".\*?(?=\")|\[[\w\*?,?\s\*]\*?\]|(?=\[\s\{))|\})')  
a = re.findall(key\_znach\_pattern, json\_text)  
for i in a:  
 key\_znach = re.split(r'\":\s\*(?:\"|\[)', i)  
 if key\_znach==['}']:  
 tabcount -= 1  
 xml\_file.write('\t' \* tabcount + '</' + keymassive[-1] + '>\n')  
 keymassive = keymassive[:-1]  
 elif len(key\_znach) == 1:  
 key = key\_znach[0].replace('": ', '')  
 xml\_file.write('\t' \* tabcount + '<' + key + '>\n')  
 keymassive.append(key)  
 tabcount+=1  
 elif i[-1] == ']':  
 massive = key\_znach[1][:-1].split(', ')  
 for a in massive:  
 xml\_file.write('\t' \* tabcount + '<' + key\_znach[0] + '>' + a + '</' + key\_znach[0] + '>\n')  
 else:  
 xml\_file.write('\t' \* tabcount + '<' + key\_znach[0] + '>' + key\_znach[1] + '</' + key\_znach[0] + '>\n')  
  
print(time.perf\_counter() - starttime)  
json\_file.close()  
xml\_file.close()

Дополнительное задание №3

Используя программу из обязательного задания, дополнительного задания №1 и дополнительного задания №2 сравнить стократное время выполнения парсинга + конвертации файла. Проанализировать полученные результаты и объяснить их сходство/различие.

|  |  |  |  |
| --- | --- | --- | --- |
| Номер теста | Время выполнения обязательного задания. (сек) | Время выполнения дополнительного задания №1 (сек) | Время выполнения дополнительного задания №2. (сек) |
| 1 | 0.00095 | 0.003858 | 0.001283 |
| 2 | 0.000746 | 0.004141 | 0.001362 |
| 3 | 0.000781 | 0.003141 | 0.001206 |

Если посмотреть на время выполнения программы с библиотекой, то оно значительно больше, это связно с тем, что использование библиотеки рассчитано на парсинг любых JSON файлов, а не только исходного файла, написанного для этой работы. Библиотека осуществляет проверку на правильность JSON файла и сам парсинг в XML.

Заключение

В результате выполнения лабораторной работы я научился выполнять парсинг и конвертацию файла из JSON формата в XML и понял, что использование регулярных выражений упрощает задачу отбора данных, а использование библиотеки уменьшает количество строчек кода, повышает читаемость, но также и повышает время выполнения программы