Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерных технологий

Лабораторная работа №4

**Исследование протоколов, форматов обмена информацией   
и языков разметки документов**

Вариант: 29

**Студент:** Щукин Егор Вячеславович, группа P3114  
**Преподаватель:** Рыбаков С.Д.,  
преподаватель факультета ПИиКТ

Санкт-Петербург 2023

# Содержание

[Содержание 2](#_Toc150869416)

[Постановка задания 3](#_Toc150869417)

[Основные этапы работы 4](#_Toc150869418)

[Листинг кода программы: 5](#_Toc150869419)

[Ответы на вопросы 9](#_Toc150869420)

[Заключение 11](#_Toc150869421)

# Постановка задания

1. Создать файл с расписанием в нужном формате согласно варианту;
2. Написать программу, осуществляющую парсинг и конвертацию файла согласно варианту без использования библиотек и регулярных выражений;
3. Переписать программу с использованием библиотек, сравнить результаты;
4. Переписать программу с использование регулярных выражений, сравнить результаты;
5. Переписать программу с использование формальной грамматики, сравнить результаты;
6. Сравнить время выполнения программ;
7. Переписать программу, чтобы та осуществляла конвертацию в любой другой формат.
8. Ответить на вопросы.

# Основные этапы работы

Текст файла с расписанием в формате xml(задание 1):

<?xml version="1.0" encoding="UTF-8" ?>

<schedules>

<schedule>

<id>251337</id>

<class\_format>Очно</class\_format>

<title>Математический анализ (продвинутый уровень)</title>

<type>

<id>1</id>

<title>Практика</title>

</type>

<campus>

<id>0</id>

<address>Кронверкский пр., д.49, лит.А</address>

</campus>

<time>

<period>1</period>

<from\_time>08:20</from\_time>

<to\_time>09:50</to\_time>

</time>

<auditory>

<id>429</id>

<name>ауд. 2433 (бывш. 429)</name>

</auditory>

<teacher>

<id>285647</id>

<name>Попов Арсений Михайлович</name>

</teacher>

</schedule>

……

</schedules>

В полном файле присутствует ещё 4 пары и описания к ним сделаные по этому образцу.

# Листинг кода программы:

*Задание №2.*

def json1(fileIn,fileout):  
 f = open(fileIn, "r", encoding="UTF-8")  
 a = []  
 a = []  
 while True:  
 line=(f.readline())  
 a.append(line)  
 if not line:  
 break  
 f.close()

a[0] = "{"  
 for i in range(1, len(a) - 1):  
 for j in range(len(a[i]) - 1):  
 if a[i][j] == "<" and a[i][j:j + 2] != "</":  
 o1 = 0  
 for z in range(j, len(a[i])):  
 if a[i][z] == ">":  
 name = a[i][j + 1:z]  
 summ = z - (j + 1)  
 o1 = z - 1  
 break  
 if i == 1:  
 a[i] = 4 \* " " + '"' + name + '": ['  
 break  
 if summ + j + 3 == len(a[i]) and len(a[i][0:j]) != 4:  
 a[i] = 4 \* " " + a[i][0:j] + '"' + name + '": {'  
 break  
 if summ + j + 3 == len(a[i]) and len(a[i][0:j]) == 4:  
 a[i] = 4 \* " " + a[i][0:j] + '{'  
 break  
 elif a[i + 1][j - 4:j - 2] == "</":  
 if a[i][(j + summ + 2):(len(a[i]) - summ - 4)].isdigit():  
 a[i] = 4 \* " " + a[i][0:j] + '"' + name + '": ' + a[i][(j + summ + 2):(len(a[i]) - summ - 4)]  
 else:  
 a[i] = 4 \* " " + a[i][0:j] + '"' + name + '": "' + a[i][(j + summ + 2):(len(a[i]) - summ - 4)] + '"'  
  
 break  
 else:  
 if a[i][(j + summ + 2):(len(a[i]) - summ - 4)].isdigit():  
 a[i] = 4 \* " " + a[i][0:j] + '"' + name + '": ' + a[i][(j + summ + 2):(len(a[i]) - summ - 4)] + ","  
 else:  
 a[i] = 4 \* " " + a[i][0:j] + '"' + name + '": "' + a[i][  
 (j + summ + 2):(len(a[i]) - summ - 4)] + '",'  
 break  
 if len(a[i][0:j]) == 0 and a[i][j:j + 2] == "</":  
 a[i] = 4 \* " " + "]"  
 break  
 elif a[i][j:j + 2] == "</" and a[i + 1][j - 4:j - 2] == "</":  
 a[i] = 4 \* " " + a[i][0:j] + "}"  
 break  
 elif a[i][j:j + 2] == "</" and a[i + 1][j - 4:j - 2] != "</":  
 a[i] = 4 \* " " + a[i][0:j] + "},"  
 break  
  
 f = open(fileout, "w", encoding="UTF-8")  
 for i in range(len(a)):  
 line = a[i] + "\n"  
 f.write(line)  
 f.write("}")  
 f.close()  
json1("Friday.xml","jsonFriday.json")

*Задание №3.*

import xmltodict  
import json  
def json2(filein,fileout):  
 with open(filein,"r",encoding="UTF-8") as f:  
 per=xmltodict.parse(f.read())  
 line=json.dumps(per,ensure\_ascii=False)  
 with open(fileout,"w",encoding="UTF-8") as fd:  
 fd.write(line)  
json2("Friday.xml","jsonFriday.json")

*Задание №4.*

import re  
def json3(filein, fileout):  
 f = open(filein, "r", encoding="UTF-8")  
 a = []  
 while True:  
 line=(f.readline())  
 a.append(line)  
 if not line:  
 break  
  
 f.close()

a[0] = "{"  
 for i in range(1, len(a) - 1):  
 for j in range(len(a[i]) - 1):  
 if re.fullmatch(r"<", a[i][j]) and a[i][j:j + 2] != "</":  
 for z in range(j, len(a[i])):  
 if re.match(r">", a[i][z]):  
 name = a[i][j + 1:z]  
 summ = z - (j + 1)  
 break  
 if i == 1:  
 a[i] = 4 \* " " + '"' + name + '": ['  
 break  
 if summ + j + 3 == len(a[i]) and len(a[i][0:j]) != 4:  
 a[i] = 4 \* " " + a[i][0:j] + '"' + name + '": {'  
 break  
 if summ + j + 3 == len(a[i]) and len(a[i][0:j]) == 4:  
 a[i] = 4 \* " " + a[i][0:j] + '{'  
 break  
 elif re.fullmatch(r"</", a[i + 1][j - 4:j - 2]):  
 if re.match(r"^[0-9]+$", a[i][(j + summ + 2):(len(a[i]) - summ - 4)]):  
 a[i] = 4 \* " " + a[i][0:j] + '"' + name + '": ' + a[i][(j + summ + 2):(len(a[i]) - summ - 4)]  
 else:  
 a[i] = 4 \* " " + a[i][0:j] + '"' + name + '": "' + a[i][  
 (j + summ + 2):(len(a[i]) - summ - 4)] + '"'  
  
 break  
 else:  
 if re.fullmatch(r"^[0-9]+$", a[i][(j + summ + 2):(len(a[i]) - summ - 4)]):  
 a[i] = 4 \* " " + a[i][0:j] + '"' + name + '": ' + a[i][  
 (j + summ + 2):(len(a[i]) - summ - 4)] + ","  
 else:  
 a[i] = 4 \* " " + a[i][0:j] + '"' + name + '": "' + a[i][  
 (j + summ + 2):(len(a[i]) - summ - 4)] + '",'  
 break  
 if len(a[i][0:j]) == 0 and re.match(r"</", a[i][j:j + 2]):  
 a[i] = 4 \* " " + "]"  
 break  
 elif re.match(r"</", a[i][j:j + 2]) and re.match(r"</", a[i + 1][j - 4:j - 2]):  
 a[i] = 4 \* " " + a[i][0:j] + "}"  
 break  
 elif re.match(r"</", a[i][j:j + 2]) and a[i + 1][j - 4:j - 2] != "</":  
 a[i] = 4 \* " " + a[i][0:j] + "},"  
 break  
 f = open(fileout, "w", encoding="UTF-8")  
 for i in range(len(a)):  
 line = a[i] + "\n"  
 f.write(line)  
 f.write("}")  
 f.close()  
json3("Friday.xml","jsonFriday.json")

*Задание №6.*

print(timeit('json1("Friday.xml","jsonFriday.json")',globals={"json1": json1,},number=10000,))  
print(timeit('json2("Friday.xml","jsonFriday.json")',globals={"json2": json2,},number=10000,))  
print(timeit('json3("Friday.xml","jsonFriday.json")',globals={"json3": json3,},number=10000,))

Вывод:  
My: 13.418502799991984/10000=0,00134185028

Lib: 9.537647000004654/10000=0,0009537647

Re:42.615107299992815/10000=0,00426151073  
Почему так, понять не сложно: собственный метод не оптимизирован и может проверять на лишнее + большое количество циклов для проверки плохо сказывается на скорости, а также есть условные конструкции, а библиотека тоже делает лишние проверки, но они там оптимизированы больше, чем в моем коде. А метод с регулярными выражениями работает медленно, потому что сами регулярные выражения достаточно медленные.

*Задание №5*

import re  
def xml\_parse(input\_file: str):  
 xml\_text = ''.join(open(input\_file, encoding='utf-8').readlines()[0:])  
 return xml\_text  
def parse\_part(a):  
 count=1  
 line=a  
 i=0  
 tag=""  
 answer=[]  
 while line!="":  
 tag = re.search(r'<[^>]\*>', line)  
 start = tag.start()  
 end = tag.end()  
 tag1 = line[start+1:end-1]  
 if tag1!='?xml version="1.0" encoding="UTF-8" ?':  
 if not re.fullmatch(r"/\*",tag1) and tag1[0]!="/":  
 tag=re.search(r'<[^>]\*>', line[end+1:(len(line))])  
 if tag!='None':  
 start1 = tag.start()  
 end1= tag.end()  
 tag2 = line[end+1+start1+1:end1+end]  
 if count==1:  
 answer.append(" "\*4\*count+'"'+tag1+'": [ ')  
 line=line[end:len(line)]  
 count+=1  
 elif count>=3 and (tag2=="/"+tag1 or tag1=="/"+tag2) :  
 if re.fullmatch(r"^[0-9]+$",line[end:end+start1+1]):  
 answer.append(" "\*4\*count+'"'+tag1+'": '+line[end:end+1+start1]+',')  
 else:  
 answer.append(" "\*4\*count+'"'+tag1+'": "'+line[end:end+1+start1]+'",')  
 line=line[end1+end:len(line)]  
 elif count>=3 and (tag2!="/"+tag1 or tag1!="/"+tag2):  
 answer.append(" "\*4\*count+'"'+tag1+'": {')  
 line=line[end:len(line)]  
 count+=1  
 elif count==2 and (tag2!="/"+tag1 or tag1!="/"+tag2):  
 answer.append(" "\*4\*count+'{')  
  
 line=line[end:len(line)]  
 count+=1  
 else:  
 if count==2:  
 answer.append(" "\*4\*count+']')  
 answer[-2]=answer[-2][0:len(answer[-2])-1]  
 line=line[end:len(line)]  
 count-=1  
 elif count>=2:  
 answer.append(" "\*4\*count+'},')  
 answer[-2]=answer[-2][0:len(answer[-2])-1]  
 line=line[end:len(line)]  
 count-=1  
 else:  
 line = line[end:len(line)]  
 answer.append("{")  
 answer.append("}")  
 return (answer)  
array=xml\_parse("person.xml")  
array=parse\_part(array)  
for i in range(len(array)):  
 print(array[i])

# Ответы на вопросы

Что более компактно? Какие форматы?

Почему используют xml до сих пор?

Для чего нужны языки разметок?

1. В чём разница между Markup и Markdown?  
   Markup – это любой язык разметки, markdown – упрощение на html, позволяющее проще и аккуратнее осуществлять некоторое форматирование.
2. В чём заключается особенность PROTOBUF по сравнению с другими форматами?  
   Хранение информации в двоичном коде, что позволяет гораздо быстрее её передавать и тратить значительно меньше памяти в сравнение с другими языками вёрстки.
3. Чем формат CSV отличается от формата TSV?  
   TSV не использует запятую как разделитель данных, что является плюсом, т.к. запятые встречаются достаточно часто.
4. Чем обусловлено постоянное появление новых форматов представления данных?  
   Всегда появляются новые задачи, для которых более удобным будет тот или иной (иногда новый) тип данных.
5. Каким образом в формате XML представляются символы ‘>’ и ‘<’?  
   При помощи спец. символов &lt; и &gt;
6. Что такое сериализация данных?  
   Процесс перевода структуры данных в формат, пригодный для хранения и передачи.
7. Каким образом в YAML обозначаются комментарии?  
   Символом # в начале строки.
8. Пояснить, как в языке разметки Markdown создать заголовки разных уровней, оформить код, вывести полужирный, курсивный и зачеркнутый текст?  
   #заголовок, ##заголовок 2 (и т.д.), \*курсив\*, \*\*жирный текст\*\*, ~~зачёркнутый~~,`вставка кода`, ```блок кода```
9. Какие форматы обмена данных используются в современных популярных мессенджерах (Viber, WhatsApp, Telegram и т.д.)?  
   json
10. Как расшифровывается абревиатура SVG?  
    Scalsble Vector Graphics (формат векторной графики)
11. Привести пример использования в языке HTML тега, который создаёт гиперссылку на url.  
    <a href=”адрес ссылки”>Ссылка</a>
12. Какое число структур может представлять собой в закодированном виде JSON-текст?  
    2 (объекты и массивы)

# Заключение

В ходе работы мною были изучены принципы работы популярных языков разметки(json, xml, yaml, html) и их парсеров. Также был написанный собственный парсер с xml на json.