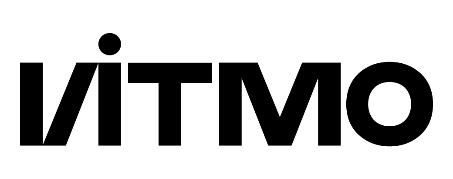
**федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»**

****

Факультет ПИиКТ

Лабораторная работа №4, вариант 3

Выполнил:  
Берман Денис Константинович P3133

Принял:

Балакшин Павел Валерьевич

Санкт-Петербург

2022

Оглавление

[Задание 3](#_Toc116056690)

[Основные этапы решения: 3](#_Toc116056691)

[Вывод: 5](#_Toc116056692)

[Список литературы: 5](#_Toc116056693)

Задание:

**Основное задание**

a) Написать программу на языке Python 3.x, которая бы осуществляла парсинг и конвертацию исходного файла в новый(YAML – JSON).

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

**Дополнительное задание №1:**

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**

a) Переписать исходный код, добавив в него использование

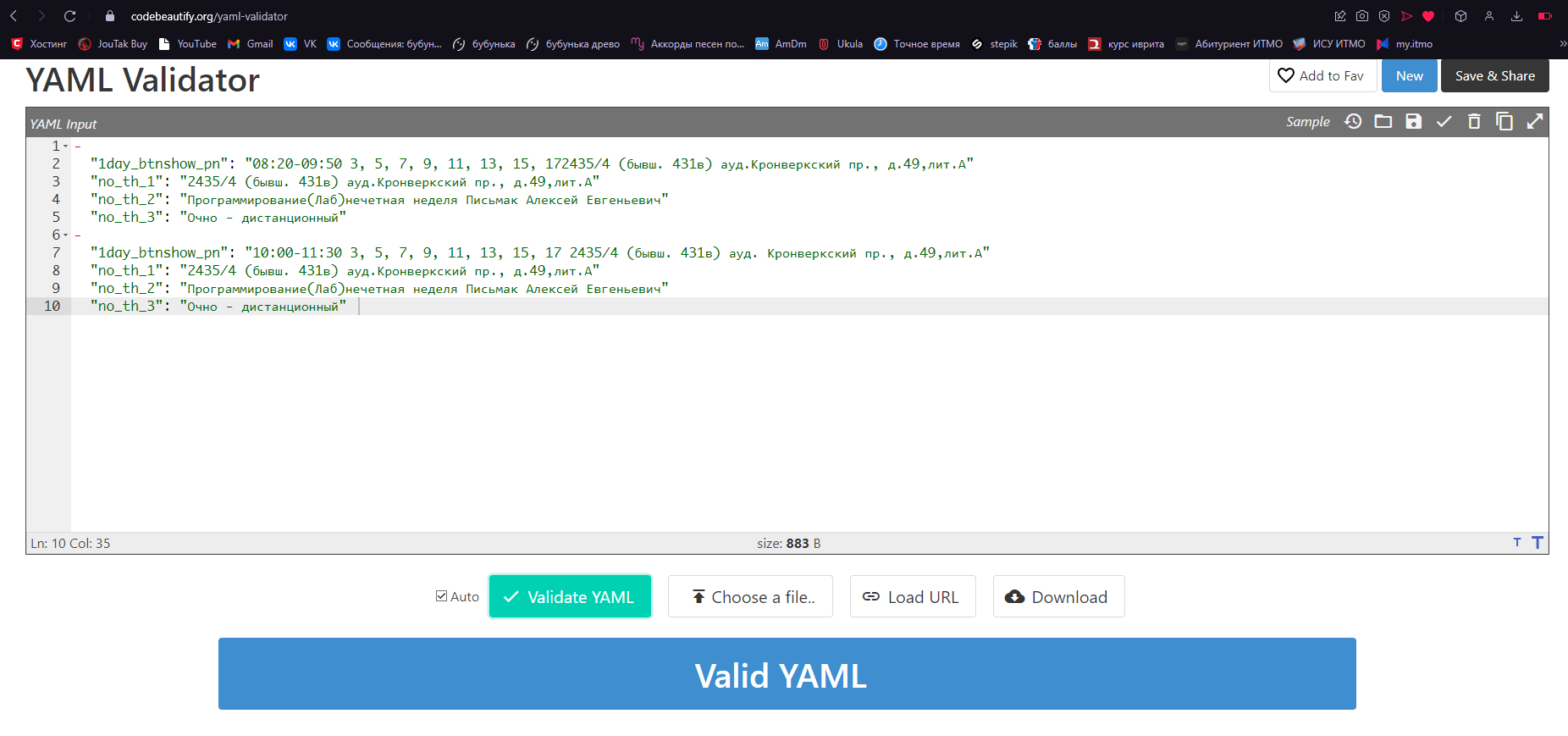
регулярных выражений.

1. b) Сравнить полученные результаты и объяснить их сходство/различие.
2. **Дополнительное задание №3**
3. 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.
4. b) Проанализировать полученные результаты и объяснить их сходство/различие.

## Основные этапы решения:

1.

a) Создадим исходный файл и докажем, что он валидный с помощью онлайн валидатора:



Напишем функцию, осуществляющую парсинг:

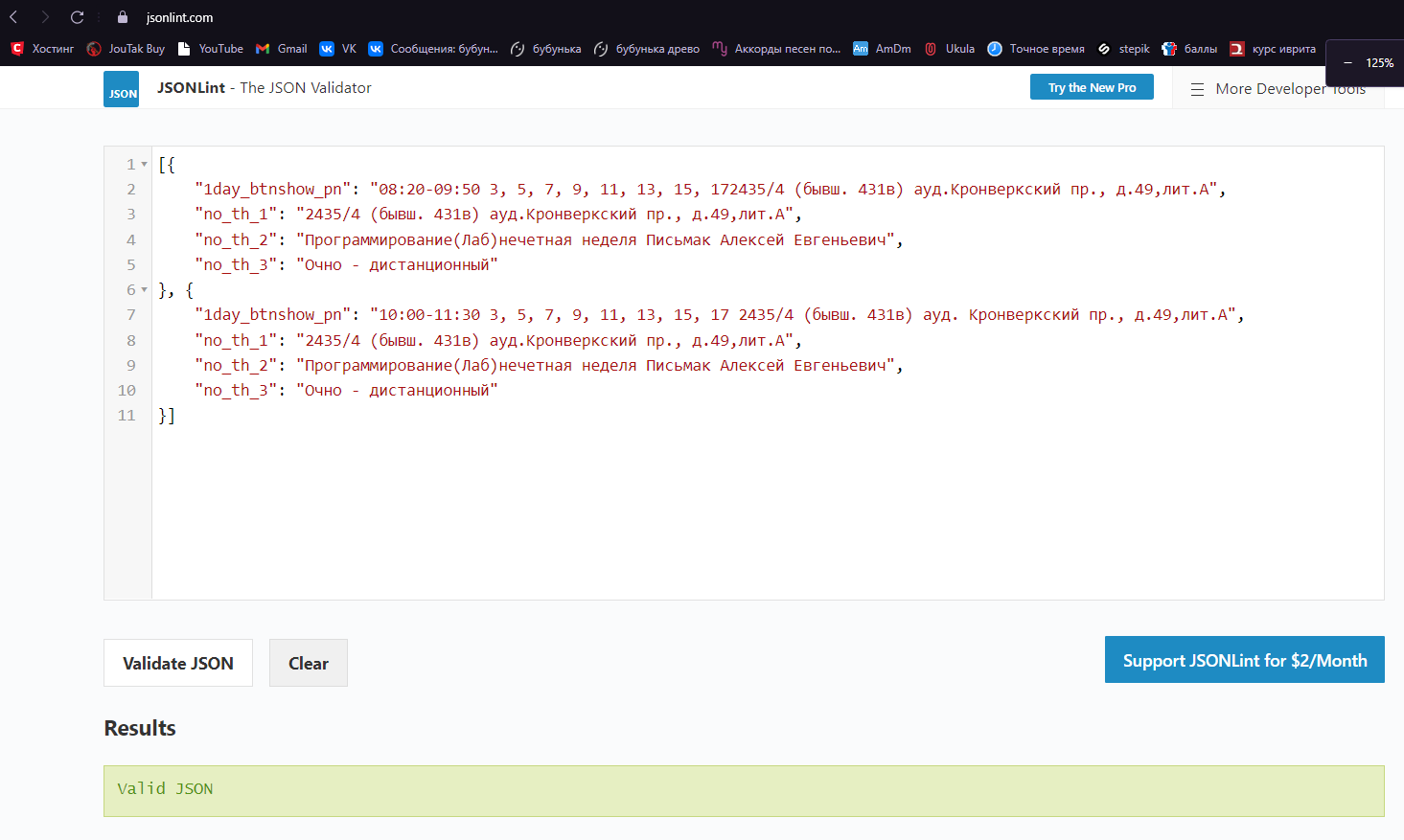
def first(x):  
 x=list(x.split('\*'))  
 flag=0;flags=['{',' },{',]  
 x[0]='['+x[0];x[-1]=x[-1]+'\n}]'  
 for i in range(len(x)-1):  
 if '- \n' in x[i]:  
 x[i]=x[i].replace('-',flags[flag],1)  
 flag=1  
 if "\"" in x[i+1] and not '{' in x[i]:  
 x[i]=x[i][:-1]+',\n'  
 return x

она будет подгонять синтаксис нашего YAML под синтаксис JSON.

Также нам пригодится функция, которая будет записывать готовые json’ы в файлы:

def printext(s,n):  
 with open(str(n)+'.json','w+') as f:  
 f.write(''.join(s))  
 print(str(n)+'.json записан способом №'+str(n)+'!')

получился такой файл, скопируем и также проверим его валидность:



Значит парсер работает!

2

Скачаем библиотеку PyYaml, и напишем процедуру для второго задания:

def second(s):  
 import yaml  
 import json  
 YML=s  
 with open(YML) as yamlin,open('2.json','w') as jsonout:  
 yamlsafe=yaml.safe\_load(yamlin)  
 json.dump(yamlsafe,jsonout,ensure\_ascii=False)  
 print('2.json записан способом №2!')#код выглядит ощутимо легче

3

Так же с использованием регулярных выражений (для вывода будем использовать функцию из 1 задания):

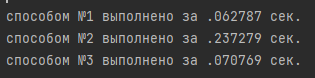
def third(s):  
 import re  
 s=''.join(s)  
 regex1=r'-\s\n(.|\n(?!-\s\n))+'  
 regex2=r'\S\n(?!})'  
 reg=re.findall(regex1,s)  
 stri='[{'+'}+,{'.join(reg)+'}]'  
 stri=re.sub(regex2,',\n',stri)  
 return stri

4

С помощью цикла “for" запустим нашу функцию100 раз, засекая время:

t1s=datetime.now()  
for i in range(100):  
 frst = first('\*'.join(s))  
 printext(frst, 1)  
t1e=datetime.now();t1d=t1e-t1s #на примере первой функции

выведем итоговую разницу во времени между стартом и концом цикла:



## Вывод:

Быстрее всего с парсингом YAML в JSON в моём случае справляется обычный код, но он заточен только под конкретный случай и может не работать с другими вариациями синтаксиса. Библиотекой же получилось дольше всего, но библиотека покрывает гораздо больше случаев и работать с ней проще.

## Список литературы:

1. Телеграмм канал Балакшина П.В. https://t.me/balakshin\_students
2. YAML Validator - <https://codebeautify.org/yaml-validator>
3. JSON Validator - https://jsonlint.com