Санкт-Петербургский Национальный Исследовательский  
Университет ИТМО

Факультет программной инженерии и компьютерной техники

**Лабораторная работа №4**

По информатике

Вариант 3

Выполнил:

Студент группы P3117

Васильченко Роман Антонович

Преподаватель:

Ильина Аглая Геннадьевна



Санкт-Петербург

2021

Оглавление

[Задание 2](#_Toc86175128)

[Основные этапы вычисления 3](#_Toc86175129)

[getHTML.py 3](#_Toc86175130)

[StartJson.py 3](#_Toc86175131)

[json2yaml.py 9](#_Toc86175132)

[json2yamlWithLibrary.py 10](#_Toc86175133)

[getHTMLwithBS4.py 10](#_Toc86175134)

[compareJson2Yaml.py 11](#_Toc86175135)

[json2YamlRegex.py 11](#_Toc86175136)

[PythonJson2Yaml.yaml 16](#_Toc86175137)

[json2csv.py 18](#_Toc86175138)

[PythonJson2CSV.csv 19](#_Toc86175139)

[Вывод 23](#_Toc86175140)

# Задание

№ Варианта: 3 --> JSON 🡪 YAML (Понедельник)

1. 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.
2. Нельзя использовать готовые библиотеки, в том числе регулярные выражения в Python и библиотеки для загрузки XML-файлов.
3. Дополнительное задание задание No1 (позволяет набрать +10 процентов от максимального числа баллов БаРС за данную лабораторную).
   1. a)  Найти готовые библиотеки, осуществляющие аналогичный парсинг и конвертацию файлов.
   2. b)  Переписать исходный код, применив найденные библиотеки. Регулярные выражения также нельзя использовать.
   3. c)  Сравнить полученные результаты и объяснить их сходство/различие.
4. Дополнительное задание задание No2 (позволяет набрать +10 процентов от максимального числа баллов БаРС за данную лабораторную).
   1. a)  Переписать исходный код, добавив в него использование регулярных выражений.
   2. b)  Сравнить полученные результаты и объяснить их сходство/различие.

5.Дополнительное задание задание No3 (позволяет набрать +10 процентов от максимального числа баллов БаРС за данную лабораторную).

1. a)  Используя свою исходную программу из обязательного задания, программу из дополнительного задания No1 и программу из дополнительного задания No2, сравнить десятикратное время выполнения парсинга + конвертации в цикле.
2. b)  Проанализировать полученные результаты и объяснить их сходство/различие.

6.Дополнительное задание задание No4 (позволяет набрать +5 процентов от максимального числа баллов БаРС за данную лабораторную.

1. a)  Переписать исходную, чтобы она осуществляла парсинг и конвертацию исходного файла в любой другой формат (кроме JSON, YAML, XML, HTML): PROTOBUF, TSV, CSV, WML и т.п.
2. b)  Проанализировать полученные результаты, объяснить осоебнности использованного формата.

# Основные этапы вычисления

## getHTML.py

import requests

url = "https://itmo.ru/ru/schedule/0/P3109/schedule.htm"

page = requests.get(url)

html = page.content.decode("utf-8")

startIndex = html.index('<table id="1day"')

endIndex = html.index('</table>', startIndex)

html = html[startIndex:endIndex+8]

html = "<div>" + html + "</div>"

## StartJson.py

{

"table": {

"id": "1day",

"tbody": [

{

"tr": [

{

"th": {

"class": "day",

"span": "ПН"

}

},

{

"td": {

"class": "time",

"body": [

{

"span": "08:20-09:50"

},

{

"dt": "четная неделя"

},

{

"dd": "305 ауд."

},

{

"dt": {

"span": "Кронверкский пр., д.49,лит.А"

}

}

]

}

},

{

"td": {

"class": "room",

"body": {

"dl": {

"dd": "305 ауд.",

"dt": {

"span": "Кронверкский пр., д.49, лит.А"

}

}

}

}

},

{

"td": {

"class": "lesson",

"dl": [

{

"dd": "Основы профессиональной деятельности(Лаб)"

},

{

"dt": {

"b": "Блохина Елена Николаевна"

}

}

]

}

},

{

"td": {

"class": "lesson-format",

"body": "Очно - дистанционный"

}

}

]

},

{

"tr": [

{

"th": {

"class": "day"

}

},

{

"td": {

"class": "time",

"body": [

{

"span": "08:20-09:50"

},

{

"div": "3, 5, 7, 9, 11, 13, 15, 17"

},

{

"dd": "305 ауд."

},

{

"dt": {

"span": "Кронверкский пр., д.49,лит.А"

}

}

]

}

},

{

"td": {

"class": "room",

"body": {

"dl": {

"dd": "305 ауд.",

"dt": {

"span": "Кронверкский пр., д.49, лит.А"

}

}

}

}

},

{

"td": {

"class": "lesson",

"dl": [

{

"dd": "Программирование(Лаб)"

},

"нечетная неделя",

{

"dt": {

"b": "Шешуков Дмитрий Михайлович"

}

}

]

}

},

{

"td": {

"class": "lesson-format",

"body": "Очно - дистанционный"

}

}

]

},

{

"tr": [

{

"th": {

"class": "day"

}

},

{

"td": {

"class": "time",

"body": [

{

"span": "10:00-11:30"

},

{

"div": "3, 5, 7, 9, 11, 13, 15, 17"

},

{

"dd": "305 ауд."

},

{

"dt": {

"span": "Кронверкский пр., д.49,лит.А"

}

}

]

}

},

{

"td": {

"class": "room",

"body": {

"dl": {

"dd": "305 ауд.",

"dt": {

"span": "Кронверкский пр., д.49, лит.А"

}

}

}

}

},

{

"td": {

"class": "lesson",

"dl": [

{

"dd": "Программирование(Лаб)"

},

"нечетная неделя",

{

"dt": {

"b": "Шешуков Дмитрий Михайлович"

}

}

]

}

},

{

"td": {

"class": "lesson-format",

"body": "Очно - дистанционный"

}

}

]

},

{

"tr": [

{

"th": {

"class": "day"

}

},

{

"td": {

"class": "time",

"body": [

{

"span": "10:00-11:30"

},

{

"dt": "четная неделя"

},

{

"dd": "305 ауд."

},

{

"dt": {

"span": "Кронверкский пр., д.49,лит.А"

}

}

]

}

},

{

"td": {

"class": "room",

"body": {

"dl": {

"dd": "305 ауд.",

"dt": {

"span": "Кронверкский пр., д.49, лит.А"

}

}

}

}

},

{

"td": {

"class": "lesson",

"dl": [

{

"dd": "Основы профессиональной деятельности(Лаб)"

},

"четная неделя",

{

"dt": {

"b": "Блохина Елена Николаевна"

}

}

]

}

},

{

"td": {

"class": "lesson-format",

"body": "Очно - дистанционный"

}

}

]

}

]

}

}

## json2yaml.py

import json

from os import error

from enum import Enum

class BracketsType(Enum):

Element = " "

ListElement = "- "

def getContent(json, previousTabs):

global result

if type(json) == str:

result += json

elif type(json) == dict:

for keys in json.keys():

result += "\n" + previousTabs + BracketsType.Element.value + keys + ": "

getContent(json[keys], previousTabs + " ")

elif type(json) == list:

for index in range(len(json)):

try:

value = json[index]

if type(value) == dict:

keyValue = list(value.keys())[0]

result += "\n" + previousTabs + BracketsType.ListElement.value + keyValue + ": "

getContent(value[keyValue], previousTabs + " ")

else:

raise Exception

except Exception as e:

result += "\n" + previousTabs + BracketsType.ListElement.value + value

def json2yamlRun():

global result

result = "---\ntable:"

with open('StartJson.json') as json\_file:

data = json.load(json\_file)

getContent(data["table"], "")

with open("PythonJson2Yaml.yaml", 'w') as the\_file:

the\_file.writelines(result)

## json2yamlWithLibrary.py

import yaml

import json

def json2yamlWithLibraryRun():

with open('StartJson.json') as json\_file:

data = json.load(json\_file)

yaml.dump(data, open("PythonJson2YamlWithLibrary.yaml", "w"), allow\_unicode=True,

default\_flow\_style=False)

## getHTMLwithBS4.py

try:

from BeautifulSoup import BeautifulSoup

except ImportError:

from bs4 import BeautifulSoup

import requests

url = "https://itmo.ru/ru/schedule/0/P3109/schedule.htm"

page = requests.get(url)

html = page.content.decode("utf-8")

parsed\_html = BeautifulSoup(html)

print(parsed\_html.body.find('table', attrs={'id': '1day'}).text)

## compareJson2Yaml.py

import os

import time

import json2yaml

import json2yamlWithLibrary

start\_time = time.time()

for i in range(1, 10):

json2yaml.json2yamlRun()

ownSolution = time.time() - start\_time

print("--- %s seconds ---" % (ownSolution))

start\_time = time.time()

for i in range(1, 10):

json2yamlWithLibrary.json2yamlWithLibraryRun()

librarySolution = time.time() - start\_time

print("--- %s seconds ---" % (librarySolution))

print(

f"--- my own solution is {librarySolution / ownSolution} times faster that a library")

## json2YamlRegex.py

from itertools import chain

import re

import json

from os import error

from enum import Enum

class BracketsType(Enum):

Element = " "

ListElement = "- "

def getContent(json, previousTabs):

global result

if type(json) == str:

result += json

elif type(json) == dict:

for keys in json.keys():

result += "\n" + previousTabs + BracketsType.Element.value + keys + ": "

getContent(json[keys], previousTabs + " ")

elif type(json) == list:

for index in range(len(json)):

try:

value = json[index]

if type(value) == dict:

keyValue = list(value.keys())[0]

result += "\n" + previousTabs + BracketsType.ListElement.value + keyValue + ": "

getContent(value[keyValue], previousTabs + " ")

else:

raise Exception

except Exception as e:

result += "\n" + previousTabs + BracketsType.ListElement.value + value

def sequence(\*funcs):

if len(funcs) == 0:

def result(src):

yield (), src

return result

def result(src):

for arg1, src in funcs[0](src):

for others, src in sequence(\*funcs[1:])(src):

yield (arg1,) + others, src

return result

number\_regex = re.compile(

r"(-?(?:0|[1-9]\d\*)(?:\.\d+)?(?:[eE][+-]?\d+)?)\s\*(.\*)", re.DOTALL)

def parse\_number(src):

match = number\_regex.match(src)

if match is not None:

number, src = match.groups()

yield eval(number), src

string\_regex = re.compile(

r"('(?:[^\\']|\\['\\/bfnrt]|\\u[0-9a-fA-F]{4})\*?')\s\*(.\*)", re.DOTALL)

def parse\_string(src):

match = string\_regex.match(src)

if match is not None:

string, src = match.groups()

yield eval(string), src

def parse\_word(word, value=None):

l = len(word)

def result(src):

if src.startswith(word):

yield value, src[l:].lstrip()

result.\_\_name\_\_ = "parse\_%s" % word

return result

parse\_true = parse\_word("true", True)

parse\_false = parse\_word("false", False)

parse\_null = parse\_word("null", None)

def parse\_value(src):

for match in chain(

parse\_string(src),

parse\_number(src),

parse\_array(src),

parse\_object(src),

parse\_true(src),

parse\_false(src),

parse\_null(src),

):

yield match

return

parse\_left\_square\_bracket = parse\_word("[")

parse\_right\_square\_bracket = parse\_word("]")

parse\_empty\_array = sequence(

parse\_left\_square\_bracket, parse\_right\_square\_bracket)

def parse\_array(src):

for \_, src in parse\_empty\_array(src):

yield [], src

return

for (\_, items, \_), src in sequence(

parse\_left\_square\_bracket,

parse\_comma\_separated\_values,

parse\_right\_square\_bracket,

)(src):

yield items, src

parse\_comma = parse\_word(",")

def parse\_comma\_separated\_values(src):

for (value, \_, values), src in sequence(

parse\_value,

parse\_comma,

parse\_comma\_separated\_values

)(src):

yield [value] + values, src

return

for value, src in parse\_value(src):

yield [value], src

parse\_left\_curly\_bracket = parse\_word("{")

parse\_right\_curly\_bracket = parse\_word("}")

parse\_empty\_object = sequence(

parse\_left\_curly\_bracket, parse\_right\_curly\_bracket)

def parse\_object(src):

for \_, src in parse\_empty\_object(src):

yield {}, src

return

for (\_, items, \_), src in sequence(

parse\_left\_curly\_bracket,

parse\_comma\_separated\_keyvalues,

parse\_right\_curly\_bracket,

)(src):

yield items, src

parse\_colon = parse\_word(":")

def parse\_keyvalue(src):

for (key, \_, value), src in sequence(

parse\_string,

parse\_colon,

parse\_value

)(src):

yield {key: value}, src

def parse\_comma\_separated\_keyvalues(src):

for (keyvalue, \_, keyvalues), src in sequence(

parse\_keyvalue,

parse\_comma,

parse\_comma\_separated\_keyvalues,

)(src):

keyvalue.update(keyvalues)

yield keyvalue, src

return

for keyvalue, src in parse\_keyvalue(src):

yield keyvalue, src

def parse(s):

s = s.strip()

match = list(parse\_value(s))

if len(match) != 1:

raise ValueError("not a valid JSON string")

result, src = match[0]

if src.strip():

raise ValueError("not a valid JSON string")

return result

def json2yamlRun():

global result

result = "---\ntable:"

with open('StartJson.json') as json\_file:

data = str(json.load(json\_file))

data = parse(data)

getContent(data["table"], "")

with open("PythonJson2YamlRegex.yaml", 'w') as the\_file:

the\_file.writelines(result)

json2yamlRun()

## PythonJson2Yaml.yaml

---

table:

id: 1day

tbody:

- tr:

- th:

class: day

span: ПН

- td:

class: time

body:

- span: 08:20-09:50

- dt: четная неделя

- dd: 305 ауд.

- dt:

span: Кронверкский пр., д.49,лит.А

- td:

class: room

body:

dl:

dd: 305 ауд.

dt:

span: Кронверкский пр., д.49, лит.А

- td:

class: lesson

dl:

- dd: Основы профессиональной деятельности(Лаб)

- dt:

b: Блохина Елена Николаевна

- td:

class: lesson-format

body: Очно - дистанционный

- tr:

- th:

class: day

- td:

class: time

body:

- span: 08:20-09:50

- div: 3, 5, 7, 9, 11, 13, 15, 17

- dd: 305 ауд.

- dt:

span: Кронверкский пр., д.49,лит.А

- td:

class: room

body:

dl:

dd: 305 ауд.

dt:

span: Кронверкский пр., д.49, лит.А

- td:

class: lesson

dl:

- dd: Программирование(Лаб)

- нечетная неделя

- dt:

b: Шешуков Дмитрий Михайлович

- td:

class: lesson-format

body: Очно - дистанционный

- tr:

- th:

class: day

- td:

class: time

body:

- span: 10:00-11:30

- div: 3, 5, 7, 9, 11, 13, 15, 17

- dd: 305 ауд.

- dt:

span: Кронверкский пр., д.49,лит.А

- td:

class: room

body:

dl:

dd: 305 ауд.

dt:

span: Кронверкский пр., д.49, лит.А

- td:

class: lesson

dl:

- dd: Программирование(Лаб)

- нечетная неделя

- dt:

b: Шешуков Дмитрий Михайлович

- td:

class: lesson-format

body: Очно - дистанционный

- tr:

- th:

class: day

- td:

class: time

body:

- span: 10:00-11:30

- dt: четная неделя

- dd: 305 ауд.

- dt:

span: Кронверкский пр., д.49,лит.А

- td:

class: room

body:

dl:

dd: 305 ауд.

dt:

span: Кронверкский пр., д.49, лит.А

- td:

class: lesson

dl:

- dd: Основы профессиональной деятельности(Лаб)

- четная неделя

- dt:

b: Блохина Елена Николаевна

- td:

class: lesson-format

body: Очно - дистанционный

## json2csv.py

import json

from os import error, pathsep

from enum import Enum

def getContent(json, previousPath):

global pathText, messageText

if type(json) == str:

messageText += json + ",\n"

pathText += previousPath + ",\n"

elif type(json) == dict:

for keys in json.keys():

getContent(json[keys], previousPath + f" / {keys}")

elif type(json) == list:

for index in range(len(json)):

try:

value = json[index]

if type(value) == dict:

keyValue = list(value.keys())[0]

getContent(value[keyValue],

previousPath + f" / {index} / {keyValue}")

else:

raise Exception

except Exception as e:

messageText += json[index] + ",\n"

pathText += previousPath + f" / {index}" + ",\n"

def json2csvRun():

global pathText, messageText

pathText = ""

messageText = ""

with open('StartJson.json') as json\_file:

data = json.load(json\_file)

getContent(data["table"], "table")

with open("PythonJson2CSV.csv", 'w') as the\_file:

the\_file.writelines(pathText + messageText)

json2csvRun()

## PythonJson2CSV.csv

table / id,

table / tbody / 0 / tr / 0 / th / class,

table / tbody / 0 / tr / 0 / th / span,

table / tbody / 0 / tr / 1 / td / class,

table / tbody / 0 / tr / 1 / td / body / 0 / span,

table / tbody / 0 / tr / 1 / td / body / 1 / dt,

table / tbody / 0 / tr / 1 / td / body / 2 / dd,

table / tbody / 0 / tr / 1 / td / body / 3 / dt / span,

table / tbody / 0 / tr / 2 / td / class,

table / tbody / 0 / tr / 2 / td / body / dl / dd,

table / tbody / 0 / tr / 2 / td / body / dl / dt / span,

table / tbody / 0 / tr / 3 / td / class,

table / tbody / 0 / tr / 3 / td / dl / 0 / dd,

table / tbody / 0 / tr / 3 / td / dl / 1 / dt / b,

table / tbody / 0 / tr / 4 / td / class,

table / tbody / 0 / tr / 4 / td / body,

table / tbody / 1 / tr / 0 / th / class,

table / tbody / 1 / tr / 1 / td / class,

table / tbody / 1 / tr / 1 / td / body / 0 / span,

table / tbody / 1 / tr / 1 / td / body / 1 / div,

table / tbody / 1 / tr / 1 / td / body / 2 / dd,

table / tbody / 1 / tr / 1 / td / body / 3 / dt / span,

table / tbody / 1 / tr / 2 / td / class,

table / tbody / 1 / tr / 2 / td / body / dl / dd,

table / tbody / 1 / tr / 2 / td / body / dl / dt / span,

table / tbody / 1 / tr / 3 / td / class,

table / tbody / 1 / tr / 3 / td / dl / 0 / dd,

table / tbody / 1 / tr / 3 / td / dl / 1,

table / tbody / 1 / tr / 3 / td / dl / 2 / dt / b,

table / tbody / 1 / tr / 4 / td / class,

table / tbody / 1 / tr / 4 / td / body,

table / tbody / 2 / tr / 0 / th / class,

table / tbody / 2 / tr / 1 / td / class,

table / tbody / 2 / tr / 1 / td / body / 0 / span,

table / tbody / 2 / tr / 1 / td / body / 1 / div,

table / tbody / 2 / tr / 1 / td / body / 2 / dd,

table / tbody / 2 / tr / 1 / td / body / 3 / dt / span,

table / tbody / 2 / tr / 2 / td / class,

table / tbody / 2 / tr / 2 / td / body / dl / dd,

table / tbody / 2 / tr / 2 / td / body / dl / dt / span,

table / tbody / 2 / tr / 3 / td / class,

table / tbody / 2 / tr / 3 / td / dl / 0 / dd,

table / tbody / 2 / tr / 3 / td / dl / 1,

table / tbody / 2 / tr / 3 / td / dl / 2 / dt / b,

table / tbody / 2 / tr / 4 / td / class,

table / tbody / 2 / tr / 4 / td / body,

table / tbody / 3 / tr / 0 / th / class,

table / tbody / 3 / tr / 1 / td / class,

table / tbody / 3 / tr / 1 / td / body / 0 / span,

table / tbody / 3 / tr / 1 / td / body / 1 / dt,

table / tbody / 3 / tr / 1 / td / body / 2 / dd,

table / tbody / 3 / tr / 1 / td / body / 3 / dt / span,

table / tbody / 3 / tr / 2 / td / class,

table / tbody / 3 / tr / 2 / td / body / dl / dd,

table / tbody / 3 / tr / 2 / td / body / dl / dt / span,

table / tbody / 3 / tr / 3 / td / class,

table / tbody / 3 / tr / 3 / td / dl / 0 / dd,

table / tbody / 3 / tr / 3 / td / dl / 1,

table / tbody / 3 / tr / 3 / td / dl / 2 / dt / b,

table / tbody / 3 / tr / 4 / td / class,

table / tbody / 3 / tr / 4 / td / body,

1day,

day,

ПН,

time,

08:20-09:50,

четная неделя,

305 ауд.,

Кронверкский пр., д.49,лит.А,

room,

305 ауд.,

Кронверкский пр., д.49, лит.А,

lesson,

Основы профессиональной деятельности(Лаб),

Блохина Елена Николаевна,

lesson-format,

Очно - дистанционный,

day,

time,

08:20-09:50,

3, 5, 7, 9, 11, 13, 15, 17,

305 ауд.,

Кронверкский пр., д.49,лит.А,

room,

305 ауд.,

Кронверкский пр., д.49, лит.А,

lesson,

Программирование(Лаб),

нечетная неделя,

Шешуков Дмитрий Михайлович,

lesson-format,

Очно - дистанционный,

day,

time,

10:00-11:30,

3, 5, 7, 9, 11, 13, 15, 17,

305 ауд.,

Кронверкский пр., д.49,лит.А,

room,

305 ауд.,

Кронверкский пр., д.49, лит.А,

lesson,

Программирование(Лаб),

нечетная неделя,

Шешуков Дмитрий Михайлович,

lesson-format,

Очно - дистанционный,

day,

time,

10:00-11:30,

четная неделя,

305 ауд.,

Кронверкский пр., д.49,лит.А,

room,

305 ауд.,

Кронверкский пр., д.49, лит.А,

lesson,

Основы профессиональной деятельности(Лаб),

четная неделя,

Блохина Елена Николаевна,

lesson-format,

Очно - дистанционный,

# Вывод

Во время выполнения программы я изучил работу с библиотеками json, requests, bs4, itertools и yaml. Также я узнал, как записываются форматы YAML и CSV. На практике смог сделать парсер из json в yaml и получил многократный прирост в сравнение с библиотекой yaml.