Федеральное государственное 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 21

Выполнил: Зыков Андрей Алексеевич

Группа: Р3106

Проверил: Балакшин П.В.

Кандидат технических наук, доцент факультета ПИиКТ

Санкт-Петербург 2024

**Оглавление**

[Задание 3](#_Toc182655694)

[Основные этапы вычисления 5](#_Toc182655695)

[**Файл с расписанием** 5](#_Toc182655696)

[**Обязательное задание** 6](#_Toc182655697)

[**Дополнительное задание 1** 7](#_Toc182655698)

[**Дополнительное задание 2** 8](#_Toc182655699)

[**Дополнительное задание 3** 10](#_Toc182655700)

[**Дополнительное задание 4** 12](#_Toc182655701)

[**Дополнительное задание 5** 13](#_Toc182655702)

[Выводы по работе 15](#_Toc182655703)

[Список источников 16](#_Toc182655704)

# **Задание**

1. Определить номер варианта как остаток деления на 36 последних двух цифр своего идентификационного номера в ISU: например, 125598 / 36 = 26. В случае, если в оба указанных дня недели нет занятий, то увеличить номер варианта на восемь. В случае, если занятий нет и в новом наборе дней, то продолжать увеличивать на восемь.

2. Изучить форму Бэкуса-Наура.

3. Изучить основные принципы организации формальных грамматик.

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

5. Понять устройство страницы с расписанием на примере расписания лектора: <https://itmo.ru/ru/schedule/3/125598/raspisanie_zanyatiy.htm>

6. Исходя из структуры расписания конкретного дня, сформировать файл с расписанием в формате, указанном в задании в качестве исходного. При этом необходимо, чтобы хотя бы в одной из выбранных дней 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.

7. Обязательное задание (позволяет набрать до 45 процентов от максимального числа баллов БаРС за данную лабораторную): написать программу на языке Python 3.x или любом другом, которая бы осуществляла парсинг и конвертацию исходного файла в новый путём простой замены метасимволов исходного формата на метасимволы результирующего формата.

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

9. 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. 1 c) Сравнить полученные результаты и объяснить их сходство/различие. Объяснение должно быть отражено в отчёте.

10.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. Объяснение должно быть отражено в отчёте.

11. Дополнительное задание №3 (позволяет набрать +25 процентов от максимального числа баллов БаРС за данную лабораторную).

а) Переписать исходный код таким образом, чтобы для решения задачи использовались формальные грамматики. То есть ваш код должен уметь осуществлять парсинг и конвертацию любых данных, представленных в исходном формате, в данные, представленные в результирующем формате: как с готовыми библиотеками из дополнительного задания №1.

b) Проверку осуществить как минимум для расписания с двумя учебными днями по два занятия в каждом.

с) Сравнить полученные результаты и объяснить их сходство/различие. Объяснение должно быть отражено в отчёте.

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

a) Используя свою исходную программу из обязательного задания и программы из дополнительных заданий, сравнить стократное время выполнения парсинга + конвертации в цикле.

b) Проанализировать полученные результаты и объяснить их сходство/различие. Объяснение должно быть отражено в отчёте.

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

a) 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. 2

b) Проанализировать полученные результаты, объяснить особенности использования формата. Объяснение должно быть отражено в отчёте.

# **Основные этапы вычисления**

## **Файл с расписанием**

Schedule.yml

weeks: [1, 3, 5, 7, 9, 11, 13, 15]  
days:  
 tuesday:  
 lessons:  
 first:  
 beginning: 11:40  
 end: 13:10  
 name: Linear algebra  
 type: Lecture  
 teacher: Tushavin Gleb Vladimirovich  
 classroom: 2202  
 adress: Lomonosova 9A  
 second:  
 beginning: 13:30  
 end: 15:00  
 name: Linear algebra  
 type: Practice  
 teacher: Tushavin Gleb Vladimirovich  
 classroom: 4210  
 adress: Lomonosova 9A  
 thursday:  
 lessons:  
 first:  
 beginning: 11:40  
 end: 13:10  
 name: History of Russian science and technology  
 type: Practice  
 teacher: Belousov Alexander Sergeevich  
 classroom: 1410  
 adress: Kronversky prospect 49A  
 second:  
 beginning: 13:30  
 end: 15:00  
 name: History of Russian science and technology  
 type: Lecture  
 teacher: Belousov Alexander Sergeevich  
 classroom: 1410  
 adress: Kronversky prospect 49A

## **Обязательное задание**

Исходный код:

#вариант 21  
#YAML -> JSON вторник, четверг  
def yaml\_to\_json(yaml\_file, json\_file):  
 with open(yaml\_file, "r", encoding="utf8") as f:  
 file = f.readlines()  
 data = []  
 for i in range(len(file)):  
 s = file[i]  
 #print(s)  
 key, value = s.split(":", 1)  
 data.append([key, value, key.count(" ")])  
  
 spaces = 0  
 json\_string = "{"  
 for i in data:  
 key, value, sp = i[0].strip(), i[1].strip(), i[2]  
 if sp < spaces:  
 #удаление лишней запятой  
 ind = json\_string.rfind(",")  
 json\_string = json\_string[:ind] + json\_string[ind+1:]  
 #закрытие скобки  
 if spaces - sp == 2:  
 json\_string += "}, "  
 else:  
 json\_string += "} " \* ((spaces - sp) // 2 - 1) + "}, "  
 spaces = sp  
 if value == "":  
 json\_string += '"' + key + '": ' "{ "  
 elif value.isdigit() or value[0] == "[" and value[-1] == "]": #число или массив  
 json\_string += '"' + key + '": ' + value + ", "  
 else:  
 json\_string+= '"' + key + '": ' + '"' + value + '", '  
  
 if spaces > 0:  
 ind = json\_string.rfind(",")  
 json\_string = json\_string[:ind] + json\_string[ind + 1:]  
 json\_string += "} " \* (spaces // 2)  
 else:  
 json\_string += " "  
 json\_string += "}"  
  
 with open(json\_file, "w", encoding="utf8") as f1:  
 f1.write(json\_string)  
 return  
  
yaml\_to\_json("Schedule.yml", "Schedule.json")

Результат:

1. Schedule.json

{"weeks": [1, 3, 5, 7, 9, 11, 13, 15], "days": { "tuesday": { "lessons": { "first": { "beginning": "11:40", "end": "13:10", "name": "Linear algebra", "type": "Lecture", "teacher": "Tushavin Gleb Vladimirovich", "classroom": 2202, "adress": "Lomonosova 9A" }, "second": { "beginning": "13:30", "end": "15:00", "name": "Linear algebra", "type": "Practice", "teacher": "Tushavin Gleb Vladimirovich", "classroom": 4210, "adress": "Lomonosova 9A" } } }, "thursday": { "lessons": { "first": { "beginning": "11:40", "end": "13:10", "name": "History of Russian science and technology", "type": "Practice", "teacher": "Belousov Alexander Sergeevich", "classroom": 1410, "adress": "Kronversky prospect 49A" }, "second": { "beginning": "13:30", "end": "15:00", "name": "History of Russian science and technology", "type": "Lecture", "teacher": "Belousov Alexander Sergeevich", "classroom": 1410, "adress": "Kronversky prospect 49A" } } } } }

## **Дополнительное задание 1**

Исходный код:

import json  
import yaml  
  
def yaml\_to\_json(yaml\_file, json\_file):  
 with open(yaml\_file, "r", encoding="utf8") as yamlF, open(json\_file, "w", encoding="utf8") as jsonF:  
 yaml\_object = yaml.safe\_load(yamlF)  
 #print(yaml\_object)  
 s = json.dumps(yaml\_object)  
 jsonF.write(s)  
  
yaml\_to\_json("Schedule.yml", "Schedule.json")

Результат:

1. Schedule.json

{"weeks": [1, 3, 5, 7, 9, 11, 13, 15], "days": {"tuesday": {"lessons": {"first": {"beginning": 700, "end": 790, "name": "Linear algebra", "type": "Lecture", "teacher": "Tushavin Gleb Vladimirovich", "classroom": 2202, "adress": "Lomonosova 9A"}, "second": {"beginning": 810, "end": 900, "name": "Linear algebra", "type": "Practice", "teacher": "Tushavin Gleb Vladimirovich", "classroom": 4210, "adress": "Lomonosova 9A"}}}, "thursday": {"lessons": {"first": {"beginning": 700, "end": 790, "name": "History of Russian science and technology", "type": "Practice", "teacher": "Belousov Alexander Sergeevich", "classroom": 1410, "adress": "Kronversky prospect 49A"}, "second": {"beginning": 810, "end": 900, "name": "History of Russian science and technology", "type": "Lecture", "teacher": "Belousov Alexander Sergeevich", "classroom": 1410, "adress": "Kronversky prospect 49A"}}}}}

Результат работы программы отличается от результата работы программы из основного задания: готовые библиотеки преобразуют время в число, а не в строку, как в основном задании, вероятно, из-за особенностей реализации библиотек.

## **Дополнительное задание 2**

Код программы:

import re  
#вариант 21  
#YAML -> JSON вторник, четверг  
def yaml\_to\_json(yaml\_file, json\_file):  
 with open(yaml\_file, "r", encoding="utf8") as f:  
 file = f.readlines()  
 data = []  
 for i in range(len(file)):  
 s = file[i]  
 key, value = re.split(r':', s, maxsplit=1)[0], re.split(r':', s, maxsplit=1)[1]  
 #print(key, value)  
 data.append([key, value, key.count(" ")])  
  
 spaces = 0  
 json\_string = "{"  
 for i in data:  
 key, value, sp = i[0].strip(), i[1].strip(), i[2]  
 if sp < spaces:  
 # удаление лишней запятой  
 ind = json\_string.rfind(",")  
 json\_string = json\_string[:ind] + json\_string[ind + 1:]  
 # закрытие скобки  
 if spaces - sp == 2:  
 json\_string += "}, "  
 else:  
 json\_string += "} " \* ((spaces - sp) // 2 - 1) + "}, "  
 spaces = sp  
 if re.fullmatch(r'', value):  
 json\_string += '"' + key + '": ' "{ "  
 elif re.fullmatch(r'-?[0-9]+', value) or re.fullmatch(r'\[.\*\]', value): #число или массив  
 json\_string += '"' + key + '": ' + value + ", "  
 else:  
 json\_string+= '"' + key + '": ' + '"' + value + '", '  
  
 if spaces > 0:  
 ind = json\_string.rfind(",")  
 json\_string = json\_string[:ind] + json\_string[ind + 1:]  
 json\_string += "} " \* (spaces // 2)  
 else:  
 json\_string += " "  
 json\_string += "}"  
  
 with open(json\_file, "w", encoding="utf8") as f1:  
 f1.write(json\_string)  
 return  
  
yaml\_to\_json("Schedule.yml", "Schedule.json")

Результат:

1. Schedule.json

{"weeks": [1, 3, 5, 7, 9, 11, 13, 15], "days": { "tuesday": { "lessons": { "first": { "beginning": "11:40", "end": "13:10", "name": "Linear algebra", "type": "Lecture", "teacher": "Tushavin Gleb Vladimirovich", "classroom": 2202, "adress": "Lomonosova 9A" }, "second": { "beginning": "13:30", "end": "15:00", "name": "Linear algebra", "type": "Practice", "teacher": "Tushavin Gleb Vladimirovich", "classroom": 4210, "adress": "Lomonosova 9A" } } }, "thursday": { "lessons": { "first": { "beginning": "11:40", "end": "13:10", "name": "History of Russian science and technology", "type": "Practice", "teacher": "Belousov Alexander Sergeevich", "classroom": 1410, "adress": "Kronversky prospect 49A" }, "second": { "beginning": "13:30", "end": "15:00", "name": "History of Russian science and technology", "type": "Lecture", "teacher": "Belousov Alexander Sergeevich", "classroom": 1410, "adress": "Kronversky prospect 49A" } } } } }

Результат работы программы не отличается от результата, полученного в обязательном задании. Код программы отличается от исходного использованием регулярных выражений.

## **Дополнительное задание 3**

Код программы:

import re  
#вариант 21  
#YAML -> JSON вторник, четверг  
def yaml\_to\_json(yaml\_file, json\_file):  
 with open(yaml\_file, "r", encoding="utf8") as f:  
 file = f.readlines()  
 data = []  
 for i in range(len(file)):  
 s = file[i]  
 key, value = re.split(r':', s, maxsplit=1)[0], re.split(r':', s, maxsplit=1)[1]  
 #print(key, value)  
 data.append([key, value, key.count(" ")])  
  
 spaces = 0  
 json\_string = "{"  
 for i in data:  
 key, value, sp = i[0].strip(), i[1].strip(), i[2]  
 if sp < spaces:  
 # удаление лишней запятой  
 ind = json\_string.rfind(",")  
 json\_string = json\_string[:ind] + json\_string[ind + 1:]  
 # закрытие скобки  
 if spaces - sp == 2:  
 json\_string += "}, "  
 else:  
 json\_string += "} " \* ((spaces - sp) // 2 - 1) + "}, "  
 spaces = sp  
 if re.fullmatch(r'', value):  
 json\_string += '"' + key + '": ' "{ "  
 elif re.fullmatch(r'-?[0-9]+', value) or re.fullmatch(r'\[.\*\]', value) or value in ["null", "true", "false"]: #число или массив + true/false/null  
 json\_string += '"' + key + '": ' + value + ", "  
 else:  
 json\_string+= '"' + key + '": ' + '"' + value + '", '  
  
 if spaces > 0:  
 ind = json\_string.rfind(",")  
 json\_string = json\_string[:ind] + json\_string[ind + 1:]  
 json\_string += "} " \* (spaces // 2)  
 else:  
 json\_string += " "  
 json\_string += "}"  
  
 with open(json\_file, "w", encoding="utf8") as f1:  
 f1.write(json\_string)  
 return  
  
yaml\_to\_json("Schedule.yml", "Schedule.json")

Результат работы программы:

1. Schedule.json

{"weeks": [1, 3, 5, 7, 9, 11, 13, 15], "days": { "tuesday": { "lessons": { "first": { "beginning": "11:40", "end": "13:10", "name": "Linear algebra", "type": "Lecture", "teacher": "Tushavin Gleb Vladimirovich", "classroom": 2202, "adress": "Lomonosova 9A" }, "second": { "beginning": "13:30", "end": "15:00", "name": "Linear algebra", "type": "Practice", "teacher": "Tushavin Gleb Vladimirovich", "classroom": 4210, "adress": "Lomonosova 9A" } } }, "thursday": { "lessons": { "first": { "beginning": "11:40", "end": "13:10", "name": "History of Russian science and technology", "type": "Practice", "teacher": "Belousov Alexander Sergeevich", "classroom": 1410, "adress": "Kronversky prospect 49A" }, "second": { "beginning": "13:30", "end": "15:00", "name": "History of Russian science and technology", "type": "Lecture", "teacher": "Belousov Alexander Sergeevich", "classroom": 1410, "adress": "Kronversky prospect 49A" } } } } }

Результат работы программы не отличается от результата, полученного в обязательном задании, новая программа способна определять данные типов true, false, null, не требовавшиеся в обязательном задании, и записывать их не в виде строк.

## **Дополнительное задание 4**

Исходный код программы:

import time  
import Task1  
import ExtraTask1  
import ExtraTask2  
import ExtraTask3  
  
#1 задание  
start\_time = time.time()  
for i in range(100):  
 Task1.yaml\_to\_json("Schedule.yml", "Schedule.json")  
end\_time = time.time()  
print("Основ. 1 ", (end\_time - start\_time), "seconds")  
  
#1 доп задание  
start\_time = time.time()  
for i in range(100):  
 ExtraTask1.yaml\_to\_json("Schedule.yml", "Schedule.json")  
end\_time = time.time()  
print("Доп. 1 ", (end\_time - start\_time), "seconds")  
  
#2 доп задание  
start\_time = time.time()  
for i in range(100):  
 ExtraTask2.yaml\_to\_json("Schedule.yml", "Schedule.json")  
end\_time = time.time()  
print("Доп. 2 ", (end\_time - start\_time), "seconds")  
  
#3 доп задание  
start\_time = time.time()  
for i in range(100):  
 ExtraTask3.yaml\_to\_json("Schedule.yml", "Schedule.json")  
end\_time = time.time()  
print("Доп. 3 ", (end\_time - start\_time), "seconds")

Результат работы программы:

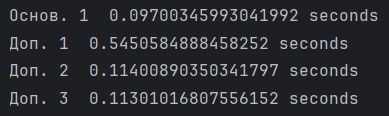


Рисунок 1 – время работы программ

Программа из обязательного задания работает быстрее всего, так как не проверяет все возможные типы данных и производит проверку строк путем использования встроенных функций и проверки отдельных символов. Программа из дополнительного задания 1 работает дольше всего, вероятно, из-за особенностей реализации готовых библиотек json и yaml. Программы из дополнительных заданий 2 и 3 работают примерно одинаковое количество времени и несколько дольше, чем программа из обязательного задания, так как осуществляют поиск и сравнение строк с использованием регулярных выражений.

## **Дополнительное задание 5**

Код программы:

def yaml\_to\_csv(yaml\_file, csv\_file):  
 with open(yaml\_file, "r", encoding="utf8") as f:  
 data = f.readlines()  
 csv\_data = {}  
 for string in data:  
 key, value = string.strip().split(":", 1)  
 value.strip()  
 if value != "":  
 if key not in csv\_data:  
 csv\_data[key] = [value]  
 else:  
 csv\_data[key].append(value)  
  
 with open(csv\_file, "w", encoding="utf8") as f:  
 for key in csv\_data:  
 string = key + ";"  
 for value in csv\_data[key]:  
 value = value[1:]  
 if value != "":  
 if value[0] == "[" and value[-1] == "]":  
 value = value[1:-1]  
 s = value.split(", ")  
 for i in s:  
 string += i + ";"  
 else:  
 string += value + ";"  
 string = string[:-1] + "\n"  
 f.write(string)  
  
  
yaml\_to\_csv("Schedule.yml", "Schedule.csv")

Результат работы программы:

1. Schedule.csv

weeks;1;3;5;7;9;11;13;15

beginning;11:40;13:30;11:40;13:30

end;13:10;15:00;13:10;15:00

name;Linear algebra;Linear algebra;History of Russian science and technology;History of Russian science and technology

type;Lecture;Practice;Practice;Lecture

teacher;Tushavin Gleb Vladimirovich;Tushavin Gleb Vladimirovich;Belousov Alexander Sergeevich;Belousov Alexander Sergeevich

classroom;2202;4210;1410;1410

adress;Lomonosova 9A;Lomonosova 9A;Kronversky prospect 49A;Kronversky prospect 49A

Программа успешно конвертирует файлы формата yaml в csv, в формате csv данные представляются в виде таблицы, где первый элемент – название строки, последующие – значения в данной строке. По умолчанию элементы разделяются точкой с запятой (или запятой), строки разделяются между собой символом переноса строки.

# **Выводы по работе**

В ходе выполнения лабораторной работы я познакомился с различными форматами обмена информацией и языками разметки документов. На практике реализовал конвертирование файлов из одного формата в другой (yaml в json, yaml в csv) и познакомился с готовыми библиотеками, реализующими аналогичный функционал, а также повторил тему предыдущей лабораторной работы – регулярные выражения.

# **Список источников**

1) Лямин А.В., Череповская Е.Н. Объектно-ориентированное программирование. Компьютерный практикум. – СПб: Университет ИТМО, 2017. – 143 с.

2) Балакшин П.В. Лекция №4. Тема: «Протоколы, форматы обмена информацией и языки разметки документов. Основы формальных грамматик.»