Федеральное государственное автономное образовательное учреждение

высшего образования «Национальный исследовательский университет

ИТМО»

*Факультет программной инженерии и компьютерной техники*

*Направление подготовки: 09.03.01 - Информатика и вычислительная техника, Компьютерные системы и технологии*

*Дисциплина «Информатика»*

**Отчет**

**По лабораторной работе №4**

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

**Вариант №7**

Выполнил:

Дворкин Борис Александрович

Группа: Р3131

Принял:

Белозубов Александр Владимирович

Г. Санкт-Петербург, 2022 г.

**Оглавление**

[Задание: 3](#_Toc118242923)

[Выполнение работы: 4](#_Toc118242924)

[Schedule.xml: 4](#_Toc118242925)

[Schedule.json: 5](#_Toc118242926)

[Schedule.yaml: 6](#_Toc118242927)

[Schedule.md: 7](#_Toc118242928)

[Сравнение моего решения и готовой библиотеки 8](#_Toc118242929)

[Сравнение моего решения с регулярными выражениями и без 8](#_Toc118242930)

[Сравнение стократного времени выполнения и конвертации. 8](#_Toc118242931)

[Конвертация исходного файла в md (markdown) 8](#_Toc118242932)

[Заключение: 9](#_Toc118242933)

[Список литературы: 10](#_Toc118242934)

# Задание:

1. 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.

2. Изучить форму Бэкуса-Наура

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

4. Понять устройство страницы с расписанием для своей группы: <http://itmo.ru/ru/schedule/0/P3110/schedule.htm>

5. 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.

6. 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

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

8. 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.

9. 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.

10. 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.

12. Проверить, что все пункты задания выполнены и выполнены верно.

# Выполнение работы:

## Исходный формат - Schedule.xml:

<timetable>

    <lang> ru </lang>

    <group> P3131 </group>

    <day> Thursday </day>

    <lessons>

        <lesson1>

            <time> 08:20-09:50 </time>

            <room> Актовый зал </room>

            <place> ул.Ломоносова, д.9, лит. М </place>

            <subject> Информатика (лекция) </subject>

            <teacher> Балакшин Павел Валерьевич </teacher>

            <lesson-format> Очно-дистанционный </lesson-format>

        </lesson1>

        <lesson2>

            <time> 10:00-11:30 </time>

            <room> Актовый зал </room>

            <place> ул.Ломоносова, д.9, лит. М </place>

            <subject> Основы профессиональной деятельности (лекция) </subject>

            <teacher> Клименков Сергей Викторович </teacher>

            <lesson-format> Очно-дистанционный </lesson-format>

        </lesson2>

        <lesson3>

            <time> 11:40-13:10 </time>

            <room> - </room>

            <place> - </place>

            <subject> Программирование (лекция) </subject>

            <teacher> Письмак Алексей Евгеньевич </teacher>

            <lesson-format> Дистанционный </lesson-format>

        </lesson3>

    </lessons>

</timetable>

## Результирующий формат - Schedule.json:

{

    "timetable": {

        "lang": "ru",

        "group": "P3131",

        "day": "Thursday",

        "lessons": {

            "lesson1": {

                "time": "08:20-09:50",

                "room": "Актовый зал",

                "place": "ул.Ломоносова, д.9, лит. М",

                "subject": "Информатика (лекция)",

                "lesson-format": "Очно-дистанционный"

            },

            "lesson2": {

                "time": "10:00-11:30",

                "room": "Актовый зал",

                "place": "ул.Ломоносова, д.9, лит. М",

                "subject": "Основы профессиональной деятельности (лекция)",

                "lesson-format": "Очно-дистанционный"

            },

            "lesson3": {

                "time": "-",

                "room": "-",

                "place": "ул.Ломоносова, д.9, лит. М",

                "subject": "Программирование (лекция)",

                "lesson-format": "Дистанционный"

            }

        }

    }

}

## Schedule.yaml:

timetable:

  lang: ru

  group: P3131

  day: Thursday

  lessons:

    lesson1:

      time: 08:20-09:50

      room: Актовый зал

      place: ул.Ломоносова, д.9, лит. М

      subject: Информатика (лекция)

      teacher: Балакшин Павел Валерьевич

      lesson-format: Очно-дистанционный

    lesson2:

      time: 10:00-11:30

      room: Актовый зал

      place: ул.Ломоносова, д.9, лит. М

      subject: Основы профессиональной деятельности (лекция)

      teacher: Клименков Сергей Викторович

      lesson-format: Очно-дистанционный

    lesson3:

      time: 11:40-13:10

      room: -

      place: -

      subject: Программирование (лекция)

      teacher: Письмак Алексей Евгеньевич

      lesson-format: Дистанционный

## Результирующий формат 2 - Schedule.md:

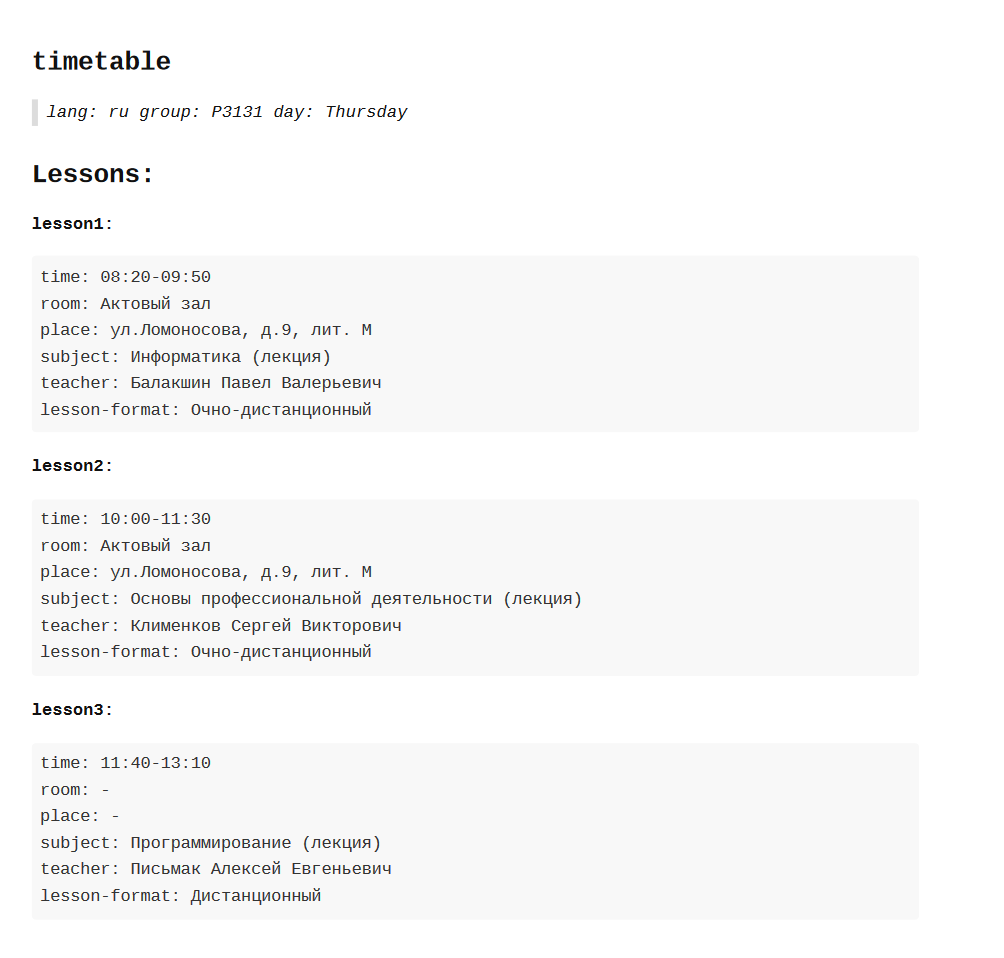


Рисунок 1

## Код xml\_to\_json.py

import time  
start\_time = time.perf\_counter()  
  
for x in range(100):  
 schedule = open('Schedule.xml', 'r', encoding="utf-8")  
 timetable = open('timetable.json', 'w', encoding="utf-8")  
  
 tab\_counter = 1  
 timetable.write('{\n')  
 x = schedule.readlines()  
 schedule.close()  
  
 for i in range(len(x)):  
 tag = x[i].strip().replace('<', '>').split('>')[1:-1]  
 if i == len(x) - 1 or x[i + 1].strip()[:2] == '</':  
 q = ''  
 else:  
 q = ',' # comma if quotes aren't closing  
 if len(tag) > 2:  
 tag.pop() # remove closing tag  
 if len(tag) == 1:  
 if tag[0][0] == '/': # if it is closing tag  
 timetable.write('\t' \* tab\_counter + '}' + q + '\n') # close, then <=> }, or }  
 tab\_counter -= 1 # we collapsed => move 1 tab left  
 else:  
 timetable.write('\t' \* tab\_counter + '"' + tag[0] + '": {\n') # open, then <=> "object": {  
 tab\_counter += 1 # we expanded => move 1 tab right  
 else:  
 timetable.write('\t' \* tab\_counter + '"' + tag[0] + '" : "' + tag[1] + '" ' + q + '\n')  
  
 timetable.write('}\n')  
 timetable.close()  
print(time.perf\_counter() - start\_time)

## Код lib\_xml\_to\_json.py

import json  
import xmltodict  
import time  
  
start\_time = time.perf\_counter()  
for x in range(100):  
 with open("Schedule.xml", encoding="utf-8") as xml\_file:  
 data\_dict = xmltodict.parse(xml\_file.read())  
 json\_data = json.dumps(data\_dict)  
 with open("data.json", "w") as json\_file:  
 json\_file.write(json\_data)  
print(time.perf\_counter() - start\_time)

## Код xml\_to\_json\_regex.py

import time  
import re  
  
start\_time = time.perf\_counter()  
for x in range(100):  
 schedule = open('Schedule.xml', 'r', encoding="utf-8")  
 timetable = open('timetable.json', 'w', encoding="utf-8")  
  
 tab\_counter = 1  
 timetable.write('{\n')  
 x = schedule.readlines()  
 schedule.close()  
  
 for i in range(len(x)):  
 tag = re.split('>', re.sub('<', '>', x[i]))[1:-1]  
 if i == len(x) - 1 or x[i + 1].strip()[:2] == '</':  
 q = ''  
 else:  
 q = ',' # comma if quotes aren't closing  
 if len(tag) > 2:  
 tag.pop() # remove closing tag  
 if len(tag) == 1:  
 if tag[0][0] == '/': # if it is closing tag  
 timetable.write('\t' \* tab\_counter + '}' + q + '\n') # close, then <=> }, or }  
 tab\_counter -= 1 # we collapsed => move 1 tab left  
 else:  
 timetable.write('\t' \* tab\_counter + '"' + tag[0] + '": {\n') # open, then <=> "object": {  
 tab\_counter += 1 # we expanded => move 1 tab right  
 else:  
 timetable.write('\t' \* tab\_counter + '"' + tag[0] + '" : "' + tag[1] + '" ' + q + '\n')  
  
 timetable.write('}\n')  
 timetable.close()  
print(time.perf\_counter() - start\_time)

print(time.perf\_counter() - start\_time)

## Код xml\_to\_md.py

import time  
  
schedule = open('Schedule.xml', 'r', encoding="utf-8")  
timetable = open('timetable.md', 'w', encoding="utf-8")  
  
start\_time = time.perf\_counter()  
tab\_counter = 0  
grid\_counter = 2  
x = schedule.readlines()  
schedule.close()  
  
for i in range(len(x)):  
 tag = x[i].strip().replace('<', '>').split('>')[1:-1]  
 if len(tag) > 2:  
 tag.pop() # remove closing tag  
 if len(tag) == 1:  
 if tag[0][0] == '/': # if it is closing tag  
 tab\_counter = 0 # we collapsed => move 1 tab left  
 grid\_counter -= 1  
 else:  
 timetable.write('#' \* grid\_counter + ' ' + tag[0] + ':\n') # open, then <=> "object": {  
 tab\_counter = 1 # we expanded => move 1 tab right  
 grid\_counter += 1  
 else:  
 if grid\_counter == 3:  
 timetable.write('> ' + tag[0] + ':' + tag[1] + '\n')  
 else:  
 timetable.write('\t' \* tab\_counter + tag[0] + ':' + tag[1] + '\n')  
  
timetable.close()  
print(time.perf\_counter() - start\_time)

## Сравнение моего решения и готовой библиотеки

В библиотеке, в отличие от моего решения, учтены все нюансы синтаксиса XML, то есть поддерживаются все его конструкции, а также библиотека предоставляет более функциональный интерфейс. Моё решение, в свою очередь, работает быстрее, но не производит приведение типов и не переводит некоторые конструкции XML в JSON, так как оно более частное и работает только с определёнными дата-файлами.

## Сравнение моего решения с регулярными выражениями и без

Решение с регулярными выражениями работает медленнее, чем решение без них, поскольку регулярные выражения неэффективны при парсинге небольших файлов. Regex следует применять на больших объёмах данных.

## Сравнение стократного времени выполнения и конвертации.

Моё решение: 0.035488

Решение с регулярками: 0.038482

Решение с использованием библиотек: 0.043471

Самое быстрое решение – без использования каких-либо библиотек. Это связано с тем, что функционал данного решения ограниченнее чем у библиотеки, как я уже говорил выше. Регулярки же проигрывают из-за того, что парсинг происходит в пределах тегов и значений маленькой длины, где эффективность регулярных выражений не проявляется и обычные строчные методы не уступают в быстродействии.

## Конвертация исходного файла в md (markdown)

Главная отличительная особенность md – это то, что этот язык разметки предназначен для визуализации данных, а не их хранения и передачи, он используется в статьях и документациях, тогда как xml используется для хранения и передачи иинформации (например, хранение данных на сервере).

Особенность конвертации xml в md в том, что представление данных xml в формате md всё также остаётся наглядным и понятным. Такая конвертация может быть довольно удобной при написании статьи, в таком случае можно конвертировать данные с сервера в формат md и запихнуть в статью, в которой они будут очень наглядно и красиво представлены.

# Заключение:

В ходе выполнения лабораторной работы я научился работать со схемами Хэмминга (7;4) и (15;11), а также определять биты информации, переданные ошибочно, и, таким образом, исправлять повреждённые в процессе передачи сообщения.

# Список литературы:

1. Форма Бэкуса-Наура: <https://ru.wikipedia.org/wiki/Форма_Бэкуса_—_Наура>
2. Балакшин П.В, доцент факультета ПИиКТ «Презентация по информатике» <https://t.me/balakshin_students>