МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Национальный исследовательский университет ИТМО»

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И  
КОМПЬЮТЕРНОЙ ТЕХНИКИ

**ЛАБОРАТОРНАЯ РАБОТА №4**

по дисциплине

«ИНФОРМАТИКА»

Исследование протоколов,

форматов обмена информацией и языков разметки документов

Вариант №409858 % 36 + 8 = 22 + 8 = 30

***Выполнил:***Студент группы P3107  
 Чусовлянов Максим Сергеевич

***Проверил:***Балакшин Павел Валерьевич

кандидат технических наук, доцент факультета ПИиКТ

**Содержание**

[Задание 3](#_t9es25z3bxif)

[Основные этапы вычисления 5](#_5klmqzmskuve)

[Заключение 9](#_2jxsxqh)

[Список литературы 10](#_z337ya)

# **Задание**

1. Определить номер варианта как остаток деления на 36 последних двух цифр своего идентификационного номера в ISU. В случае, если в данный день недели нет занятий, то увеличить номер варианта 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 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.
7. **Обязательное задание**(позволяет набрать до 45 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 путём простой замены метасимволов исходного формата на метасимволы результирующего формата. 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. Объяснение должно быть отражено в отчёте.

1. **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.

1. **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 и 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.

1. **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.

1. **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.

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

# **Основные этапы вычисления**

## 

*Рисунок 1.1 ( расписание лектора )*

Пример файла с расписанием в формате JSON:

{

"current\_day": 6,

"current\_week": 11,

"current\_time": "12:05",

"schedule": [

{

"id": 1337,

"title": "Параллельные вычисления / Parallel Computing",

"group": "PRLCOMP 1",

"teacher": {

"id": 125598,

"name": "Балакшин Павел Валерьевич"

},

"class\_type": {

"id": 0,

"name": "Лекция”

},

"class\_format": {

"id": 2,

"name": "Очно - дистанционный"

},

"date": {

"day": 6,

"weeks": [3, 7, 9, 11, 13, 15]

},

"time": {

"class\_number": 1,

"from": "08:20",

"to": "09:50"

},

"classroom": {

"id": 1324,

"name": "1324 ауд."

},

"campus": {

"id": 0,

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

}

},

…

]}

**Обязательное задание:**

Исходный файл json:

<https://github.com/Vaneshik/VT-Labs/blob/main/informatics/lab4/data/in.json>

Исходный код:

<https://github.com/Vaneshik/VT-Labs/tree/main/informatics/lab4/main_task>

Результат:

<https://github.com/Vaneshik/VT-Labs/blob/main/informatics/lab4/data/out.xml>

**Дополнительное задание 1:**

Исходный код:

<https://github.com/Vaneshik/VT-Labs/blob/main/informatics/lab4/additional_task1/main.py>

Результат:

<https://github.com/Vaneshik/VT-Labs/blob/main/informatics/lab4/data/out1.xml>

Готовые библиотеки: стандартная библиотека Python json для парсинга JSON и dicttoxml для дампа словаря в файл xml.

Файл результата не отличается от результата обязательного задания, кроме как тем, что при использовании библиотеки элементы списков заключены в парный тег “<item>”, когда при использовании самописной реализации используется “<[KEY]\_elem>”. Код программы стал значительно проще – теперь он состоит из одной функции и чтения файлов

**Дополнительное задание 2:**

Исходный код:

<https://github.com/Vaneshik/VT-Labs/tree/main/informatics/lab4/additional_task2>

Результат:

<https://github.com/Vaneshik/VT-Labs/blob/main/informatics/lab4/data/out1.xml>

Файл результата не отличается от результата обязательного задания. Единственное изменение в коде программы - парсинг чисел и строк заменены на регулярные выражения вместо циклов.

**Дополнительное задание 3:**

Исходный код:

<https://github.com/Vaneshik/VT-Labs/blob/main/informatics/lab4/additional_task3/main.py>

Результат:

<https://github.com/Vaneshik/VT-Labs/blob/main/informatics/lab4/data/out3.xml>

Мной изначально был написан код с использованием формальных грамматик. Поэтому я смог импортировать нужные мне функции из обязательного задания. Парсер умеет конвертировать JSON в любом формате, что было протестировано мной при помощи [JSON Parsing Test Suite](https://github.com/nst/JSONTestSuite).

**Дополнительное задание 4:**

Исходный код:

<https://github.com/Vaneshik/VT-Labs/tree/main/informatics/lab4/additional_task4>

1) Время работы программы для обязательного задания:

0.9949188232421875 секунды;

2) Время работы программы для доп. задания №1 (программа использует библиотеку dicttoxml):

1.314805269241333 секунды;

3) Время работы программы для доп. задания №2 (программа использует библиотеку re):

1.7888438701629639 секунды.

Данные результаты удивили меня.

Скорее всего узким горлышком доп. задания 1 является библиотека dicttoxml. Изучив ее исходный код, мы можем понять что там существует множество дополнительных проверок ( например CDATA sections), для того чтобы максимально точно соответствовать формату xml и не допускать ошибок. В то время как код из обязательного задания просто “оборачивает” значение в тег ключа с форматированными пробелами.

Регулярные выражение работают дольше, так как не используется компиляция (re.compile) регулярных выражений и проверка регуляркой идет дольше чем итерация по строке (за O(n) ).

**Дополнительное задание 5:**

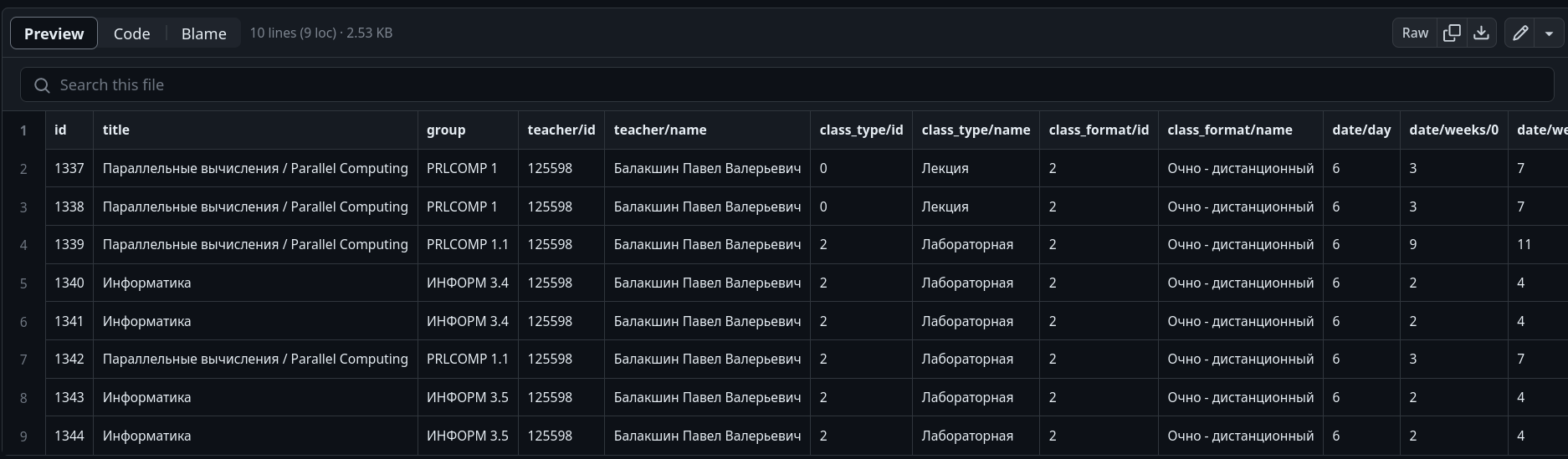
Исходный код:

<https://github.com/Vaneshik/VT-Labs/blob/main/informatics/lab4/additional_task5/main.py>

Результат:

<https://github.com/Vaneshik/VT-Labs/blob/main/informatics/lab4/data/out5.tsv>

TSV (tab separated values — значения, разделённые табуляцией) — текстовый формат для представления таблиц баз данных. Каждая запись в таблице — это строка текстового файла. Каждое поле записи отделяется от других с помощью символа табуляции, точнее горизонтальной табуляции.



*Рисунок 1.2 ( пример полученной таблицы после конвертации )*

# **Заключение**

В ходе проделанной лабораторной работы, я познакомился с разными форматами файлов. Узнал о существовании формальных грамматик и БНФ. Написал парсеры, которые конвертируют json-файл в формат xml и tsv. Получил удовольствие от 5 часового кодинга рекурсивного парсера на формальных грамматиках.

# **Список литературы**

1. [Электронный ресурс]: Википедия. Свободная энциклопедия. – Режим доступа: <https://ru.wikipedia.org/wiki/JSON>
2. Грошев А.С. Г89 Информатика: Учебник для вузов / А.С. Грошев. – Архангельск, Арханг. гос. техн. ун-т, 2010. - 470с. - Режим доступа: <http://arm.sies.uz/wp-content/uploads/2020/11/16-%D0%98%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0-2010-%D0%B4%D0%B0%D1%80%D1%81%D0%BB%D0%B8%D0%BA-%D0%93%D1%80%D0%BE%D1%88%D0%B5%D0%B2-%D0%90.%D0%A1.pdf>