МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Национальная научно-образовательная корпорация ИТМО»

Факультет Программной инженерии и компьютерной техники

Лабораторная работа №4

По дисциплине

«Информатика»

Вариант №24

**Выполнил:**

Студент группы P3114

Щербинин Эдуард Павлович

**Преподаватель:**

Машина Екатерина Алексеевна

**Содержание**

[Задание 3](#_Toc119243646)

[Основные этапы вычислений 4](#_Toc119243647)

[Вывод 7](#_Toc119243648)

[Список литературы 8](#_Toc119243649)

Задание

Вариант: 367656 % 36 = 24

JSON 🡪 XML, пятница

1. Обязательное задание: написать программу на языке Python 3.x, которая бы осуществляла парсинг и конвертацию исходного файла в новый.

2. 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.

3. Дополнительное задание №2:

a) Переписать исходный код, добавив в него использование

регулярных выражений.

1. b) Сравнить полученные результаты и объяснить их сходство/различие.
2. 4. 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.
5. 5. Дополнительное задание №4:
6. 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.
7. d) Проанализировать полученные результаты, объяснить особенности использования формата.
8. 12. Проверить, что все пункты задания выполнены и выполнены верно.

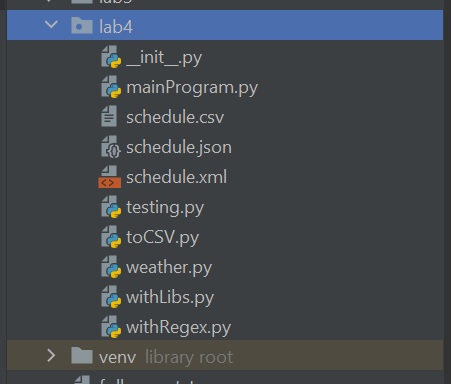
Основные этапы вычислений

Запишем исходный файл с расписанием на пятницу schedule.json:

{  
 "friday": [  
 {  
 "name": "Иностранный язык",  
 "time\_start": "8:20",  
 "time\_end": "9:50",  
 "auditory": 3225,  
 "location": "Ломоносова 9",  
 "prepod": "Сигаев Артём Сергеевич"  
 },  
 {  
 "name": "Иностранный язык",  
 "time\_start": "10:00",  
 "time\_end": "11:30",  
 "auditory": 3225,  
 "location": "Ломоносова 9",  
 "prepod": "Сигаев Артём Сергеевич"  
 },  
 {  
 "name": "Дискретная математика",  
 "time\_start": "15:20",  
 "time\_end": "16:50",  
 "auditory": 2304,  
 "location": "Кронверский пр. 49",  
 "prepod": "Поляков Владимир Иванович"  
 },  
 {  
 "name": "Дискретная математика",  
 "time\_start": "17:00",  
 "time\_end": "18:30",  
 "auditory": 2304,  
 "location": "Кронверский пр. 49",  
 "prepod": "Поляков Владимир Иванович"  
 }  
 ],  
 "null": null,  
 "int": 10,  
 "float": 10e2,  
 "bool": true,  
 "arr": [[["a"]], "b", {"key": "val"}],  
 "empty\_arr": [],  
 "obj": {}  
}

В конце добавим другие типы данных для проверки правильного парсинга файла

Для выполнения всех заданий лабораторной работы были написаны следующие программы (кроме weather.py, это программа из телеграм канала, которая должна была чем-то помочь при выполнении, но применение ей не нашлось):



*Рис 1. Полный листинг программ .py*

Про все файлы кратко:

mainProgram.py: программа, выполняющая основное задание лабы, выполняет парсинг JSON из файла schedule.json и переводит его в dict в питоне, после чего собирает в XML-строку и записывает в файл schedule.xml. Основная идея программы – рекурсия, получая на вход значение в строке, она разделяет её по внешним запятым и для каждого элемента вытаскивает пару ключ-значение, сохраняя ключ и отправляя значение (если это массив или словарь) рекурсивно в ту же функцию, пока это значение не станет строкой или другим простым типом. Таким образом восстанавливается json файл любой сложности и глубины значений. Перевод в XML работает похожим рекурсивным образом, только строка наоборот собирается.

withLibs.py: доп. задание №1, парсинг с помощью библиотек. Подключаем библиотеку json, а также dicttoxml чтобы сконвертировать в xml и парсим.

withRegex.py: доп. задание №2, парсинг с использованием регулярок, просто берём файл mainProgram.py и заменяем на регулярки всё, что получится.

\_\_init\_\_.py: его наличие позволяет импортировать функции из одних программ в другие (пустой, нужен для проведения и сравнения тестов скорости).

testing.py: доп. задание №3, непосредственно тесты скорости программ, импортируем все функции из предыдущих трёх программ и с помощью таймера замеряем время выполнения 2000 парсингов каждым способом.

toCSV.py: доп. задание №4, перевод исходного файла в другой формат (CSV). Условия невозможности использовать библиотек нет, поэтому перевод осуществлён ими.

По итогу основная программа генерирует следующую XML-строку:

<?xml version="1.0" encoding="UTF-8" ?><root><friday><name>Иностранный язык</name><time\_start>8:20</time\_start><time\_end>9:50</time\_end><auditory>3225</auditory><location>Ломоносова 9</location><prepod>Сигаев Артём Сергеевич</prepod></friday><friday><name>Иностранный язык</name><time\_start>10:00</time\_start><time\_end>11:30</time\_end><auditory>3225</auditory><location>Ломоносова 9</location><prepod>Сигаев Артём Сергеевич</prepod></friday><friday><name>Дискретная математика</name><time\_start>15:20</time\_start><time\_end>16:50</time\_end><auditory>2304</auditory><location>Кронверский пр. 49</location><prepod>Поляков Владимир Иванович</prepod></friday><friday><name>Дискретная математика</name><time\_start>17:00</time\_start><time\_end>18:30</time\_end><auditory>2304</auditory><location>Кронверский пр. 49</location><prepod>Поляков Владимир Иванович</prepod></friday><null /><int>10</int><float>1000.0</float><bool>true</bool><arr>a</arr><arr>b</arr><arr><key>val</key></arr><obj /></root>

Отступы в XML не играют роли, а только уменьшают размер файла, поэтому тут их нет, но сделав их с помощью PyCharm и сравнив с онлайн-парсерами получаем, что всё работает верно.

Вывод

Разобрался с устройством таких форматов, как JSON и XML, получил опыт в использовании рекурсии, функций и парсинга файлов в целом, провел тесты и сравнения, исходя из которых, вручную написанный парсер работает примерно в 3 раза быстрее, чем в библиотеках. Могу предположить, что это из-за того, что мой парсер более узконаправлен, чем библиотечный, и, возможно, будет работать дольше с более глубокими файлами, у которых значение в значении лежит много раз (из-за рекурсии).

Список литературы

1. Лямин А.В., Череповская Е.Н. Объектно-ориентированное программирование. Компьютерный практикум. – СПб: Университет ИТМО, 2017. – 143 с. – Режим доступа: <https://books.ifmo.ru/file/pdf/2256.pdf>. (дата обращения: 13.11.2022)
2. Википедия. Синтаксический анализ (Париснг) [Электронный ресурс] Режим доступа: <https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7> (дата обращения: 12.11.2022)