МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное Государственное Автономное Образовательное Учреждение Высшего Образования "Национальный Исследовательский Университет Итмо"

##### ФАКУЛЬТЕТ ПИиКТ

## ЛАБОРАТОРНАЯ РАБОТА №4

### по дисциплине

### «ИНФОРМАТИКА»

### Вариант № 21

##### ***Выполнил:*** Студент группы P3119 Зайцев Артём Михайлович

#### Преподаватель:

Рыбаков Степан

Дмитриевич

Санкт-Петербург, 2022

Содержание

[Задание 4](#_Toc28610)

[Основные этапы вычисления 5](#_Toc30826)

[Задание 2 5](#_Toc30460)

[Задание 3 5](#_Toc31713)

[Задание 4 5](#_Toc19348)

[Задание 5 5](#_Toc24775)

[Задание 6 5](#_Toc30202)

[Задание 7 5](#_Toc21235)

[Вывод: 6](#_Toc21889)

[Список используемой литературы 7](#_Toc29655)

# Задание

1. Понять устройство страницы с расписанием для своей группы: http://itmo.ru/ru/schedule/0/P3110/schedule.htm
2. 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

1. Написать программу на языке Python 3.x, которая бы осуществляла парсинг и конвертацию исходного файла в новый. (Нельзя использовать готовые библиотеки, в том числе регулярные выражения в Python и библиотеки для загрузки XML-файлов)
2. A) Найти готовые библиотеки, осуществляющие аналогичный

парсинг и конвертацию файлов.

B) Переписать исходный код, применив найденные

библиотеки. Регулярные выражения также нельзя

использовать.

С) Сравнить полученные результаты и объяснить их

сходство/различие.

1. A) Переписать исходный код, добавив в него использование

регулярных выражений.

B) Сравнить полученные результаты и объяснить их

сходство/различие

1. 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.

1. С) 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.

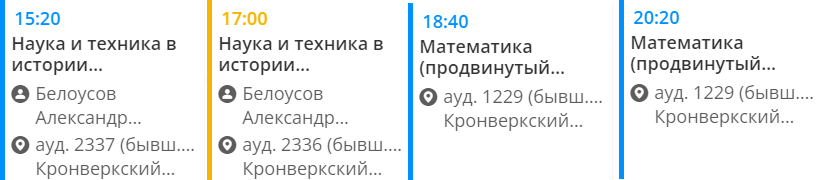
D) Проанализировать полученные результаты, объяснить

особенности использования формата

# Основные этапы вычисления

## Задание 2

День - четверг.



Файл в формате YAML: <https://github.com/nentu/inf_3/blob/main/original.yml>

## Задание 3

Код программы парсера на питоне из YAML в JSON: <https://github.com/nentu/inf_3/blob/main/task_1.py>

## Задание 4

Была найдена библиотека [yaml](https://pypi.org/project/PyYAML/)

Написана программа с использование найденной ранее библиотекой. Код программы: <https://github.com/nentu/inf_3/blob/main/task_2.py>

Вывод:

Библиотека возвращает неотформатированный для вывод json, в котором также необходимо заменить “ на ‘. После замены и форматирования результаты эквивалентны.

## Задание 5

Код программы: <https://github.com/nentu/inf_3/blob/main/task_3.py>

Результаты не отличаются, а вот программу было легче реализовывать.

## Задание 6

Код программы: <https://github.com/nentu/inf_3/blob/main/task_4_comparative.py>

Вывод программы:

100 кратное выполнения программы для 1 задания: 4.362956523895264 секунд

100 кратное выполнения программы для 2 задания: 6.857980489730835 секунд

100 кратное выполнения программы для 3 задания: 4.52534031867981 секунд

Как можно заметить, быстрее всего работает код написанный без библиотеки и регулярны выражений. На втором месте код с регулярками. Это объясняется тем, что рег. Выражения удобно ипользовать, но они медленные. Самым медленным оказался код, использующий библиотеку. Скорее всего это связано с тем, что в ней вшиты какие-либо проверки или другой функционал, не требующийся в данной лабораторно работе, но отнимающий время.

## Задание 7

Код программы реулизующий простейший конвертор из YAML в PROTOBUF. <https://github.com/nentu/inf_3/blob/main/task_5_to_PROTOBUF.py>

Результирующий файл проходит проверку на синтаксис.

Некоторые особенности формата PROTOBUF:

1. Сам файл нужен именно для упаковки данных. Поэтому каждому полю присваивается индекс, а уже во время заполнения протокола выполняется заполнение полей данными.
2. Существуют несколько модификаторов, которые должны быть у каждого поля: optional, repeated, required.
3. Для каждого ЯП есть свои правила создания PROTOBUF файла

# Вывод:

Во время выполнения лабораторной работы я познакомился с разными языками маркировки, такими как: YAML, JSON, XML, PROTOBAF. Научился реализовывать парсеры на языке Питон. Используя знания полученные ранне смог усовершенствовать программу. Уверен, что полученные знания пригодятся как в дальнейшем изучении предмета, так и в профессиональном развитии.

# Список используемой литературы

##### [Алексеев Е.Г., Богатырев С.Д. Информатика. Мультимедийный электронный учебник](http://inf.e-alekseev.ru/text/toc.html)

* [С.В. Фомин. Системы счисления](https://vk.com/doc165585709_620224544?hash=gDcpSx5TXdP7NdN7WDsf7NXPsmflxESl8q1tHzGri6k&dl=TEZL4Dq0XfJaNDx118jfT41v8bzIhfnjMqQY0vJ2TiL)