

Федеральное государственное автономное 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 №17

Выполнил: студент группы Р3108
Васильев Никита

Проверил: Балакшин Павел
Валерьевич, доцент факультета
ПИиКТ, кандидат технических наук

Санкт-Петербург 2023

Содержание

Задание.....	3
Основные этапы вычисления	4
Обязательное задание.....	4
Дополнительное задание №1	6
Сравнение полученных результатов	7
Дополнительное задание №2	7
Сравнение полученных результатов	9
Дополнительное задание №3	9
Сравнение полученных результатов	12
Дополнительное задание №4	13
Сравнение полученных результатов	14
Дополнительное задание №5	14
Заключение.....	15
Список литературы.....	17

Задание

Написать программу на языке Python 3.x, которая бы осуществляла парсинг и конвертацию исходного файла (в формате XML) в новый (в формате YAML) путём простой замены метасимволов исходного формата на метасимволы результирующего формата.

Дополнительное задание №1

1. Найти готовые библиотеки, осуществляющие аналогичный парсинг и конвертацию файлов.
2. Переписать исходный код, применив найденные библиотеки. Регулярные выражения также нельзя использовать.
3. Сравнить полученные результаты и объяснить их сходство/различие.

Дополнительное задание №2

1. Переписать исходный код, добавив в него использование регулярных выражений.
2. Сравнить полученные результаты и объяснить их сходство/различие.

Дополнительное задание №3

1. 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.
2. Проверку осуществить как минимум для расписания с двумя учебными днями по два занятия в каждом.
3. Сравнить полученные результаты и объяснить их сходство/различие.

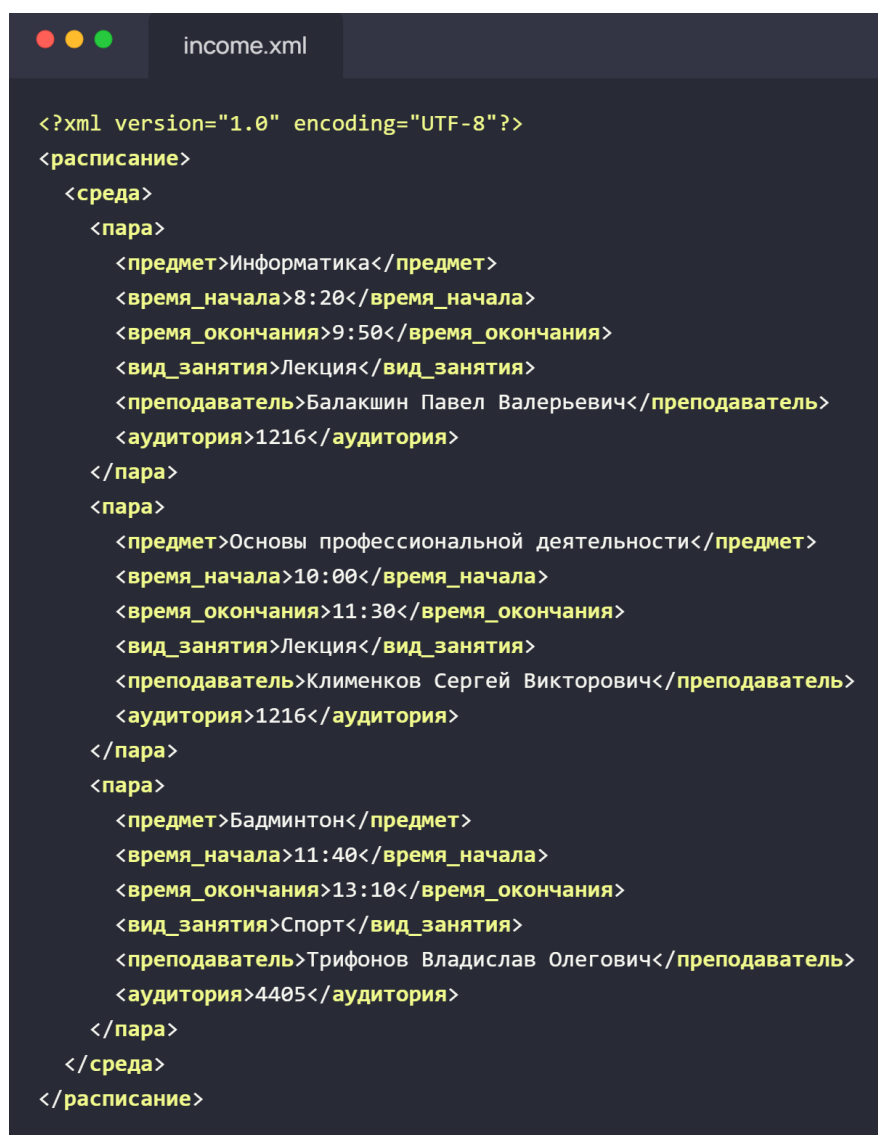
Дополнительное задание №4

1. Используя свою исходную программу из обязательного задания и программы из дополнительных заданий, сравнить стократное время выполнения парсинга + конвертации в цикле.
2. Проанализировать полученные результаты и объяснить их сходство/различие. Объяснение должно быть отражено в отчёте.

Дополнительное задание №5

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

Основные этапы вычисления



```
<?xml version="1.0" encoding="UTF-8"?>
<расписание>
  <среда>
    <пара>
      <предмет>Информатика</предмет>
      <время_начала>8:20</время_начала>
      <время_окончания>9:50</время_окончания>
      <вид_занятия>Лекция</вид_занятия>
      <преподаватель>Балакшин Павел Валерьевич</преподаватель>
      <аудитория>1216</аудитория>
    </пара>
    <пара>
      <предмет>Основы профессиональной деятельности</предмет>
      <время_начала>10:00</время_начала>
      <время_окончания>11:30</время_окончания>
      <вид_занятия>Лекция</вид_занятия>
      <преподаватель>Клименков Сергей Викторович</преподаватель>
      <аудитория>1216</аудитория>
    </пара>
    <пара>
      <предмет>Бадминтон</предмет>
      <время_начала>11:40</время_начала>
      <время_окончания>13:10</время_окончания>
      <вид_занятия>Спорт</вид_занятия>
      <преподаватель>Трифонов Владислав Олегович</преподаватель>
      <аудитория>4405</аудитория>
    </пара>
  </среда>
</расписание>
```

Рисунок 1 - Исходные данные в XML

Обязательное задание

Для конвертации из XML в YAML воспользуюсь кодом на Python, представленным на Рисунке 2.

```

main.py

def main():
    read_file = 'files/income.xml'
    write_file = 'files/outcome.yaml'
    base(read_file, write_file)

def base(read_file, write_file):
    with open(read_file, mode='r', encoding='utf-8') as xml, open(write_file, mode='w', encoding='utf-8') as yaml:

        para_cnt = 0
        for line in xml:
            start = line.find("</")
            end = line.find(">", start)
            while start != -1 and end != -1:
                line = line[:start] + line[end + 1:]

                start = line.find("</")
                end = line.find(">", start)
            line = line.replace('<', '')
            line = line.replace('>', ': ')

            if "napa" in line:
                para_cnt += 1

            if "napa" in line and para_cnt > 1:
                line = line.replace("napa:", "-")

            if "Лекция" in line:
                line = line.replace("Лекция", "Лекция'")

            if "Информатика" in line:
                line = line.replace("Информатика", "Информатика'")

            if "Основы профессиональной деятельности" in line:
                line = line.replace("Основы профессиональной деятельности", "Основы профессиональной деятельности'")

            if "Бадминтон" in line:
                line = line.replace("Бадминтон", "Бадминтон'")

            if "Спорт" in line:
                line = line.replace("Спорт", "Спорт'")

            if "Балашкин Павел Валерьевич" in line:
                line = line.replace("Балашкин Павел Валерьевич", "Балашкин Павел Валерьевич'")

            if "Клименков Сергей Викторович" in line:
                line = line.replace("Клименков Сергей Викторович", "Клименков Сергей Викторович'")

            if "Трифонов Владислав Олегович" in line:
                line = line.replace("Трифонов Владислав Олегович", "Трифонов Владислав Олегович'")

            if "8:20" in line:
                line = line.replace("8:20", "8:20'")

            if "9:50" in line:
                line = line.replace("9:50", "9:50'")

            if "10:00" in line:
                line = line.replace("10:00", "10:00'")

            if "11:30" in line:
                line = line.replace("11:30", "11:30'")

            if "11:40" in line:
                line = line.replace("11:40", "11:40'")

            if "13:10" in line:
                line = line.replace("13:10", "13:10'")

            if 'xml version="1.0" encoding="UTF-8"?': not in line:
                yaml.write(line)

        with open(write_file, 'r', encoding='utf-8') as file:
            strings = file.readlines()

            strings.insert(3, "  -" + '\n')
            non_empty = [strin for strin in strings if strin.strip() != '']

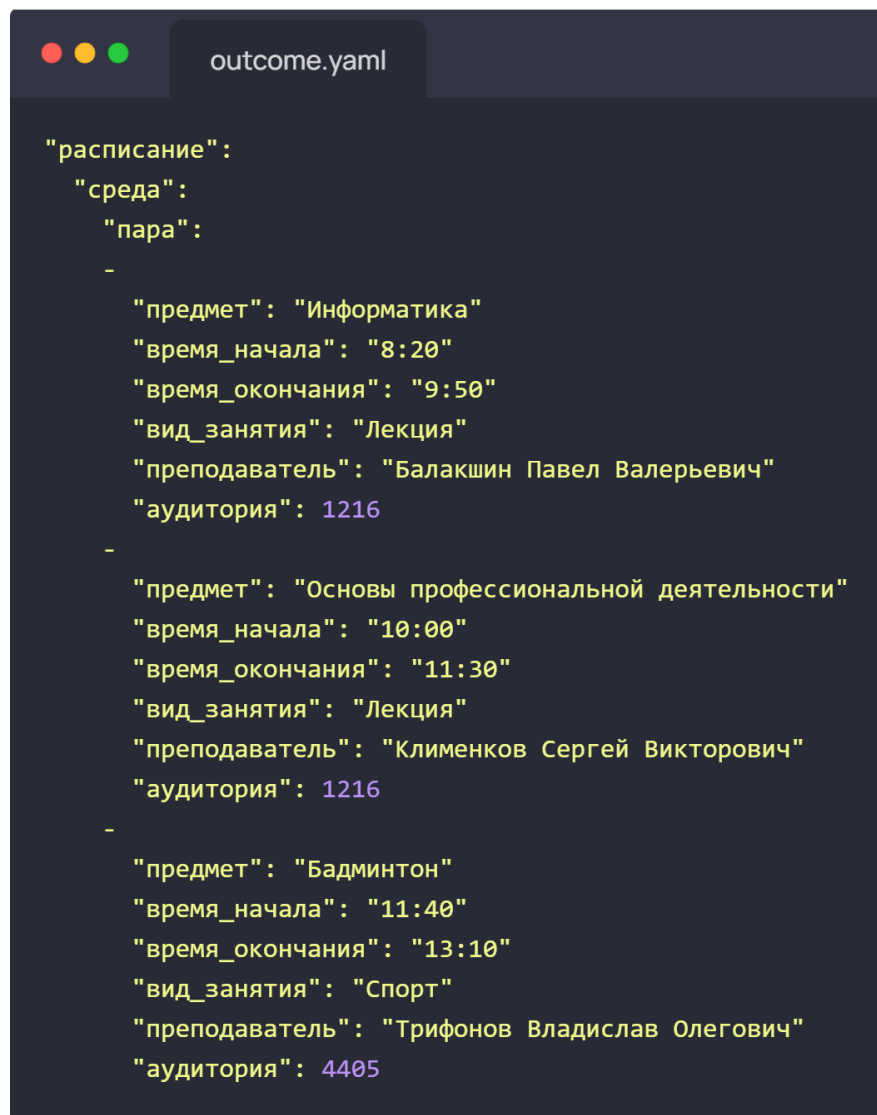
            with open(write_file, 'w', encoding='utf-8') as file:
                file.writelines(non_empty)

if __name__ == "__main__":
    main()
</pre

```

Рисунок 2 - Код на Python для основного задания

В результате получается YAML файл, представленный на Рисунке 3:

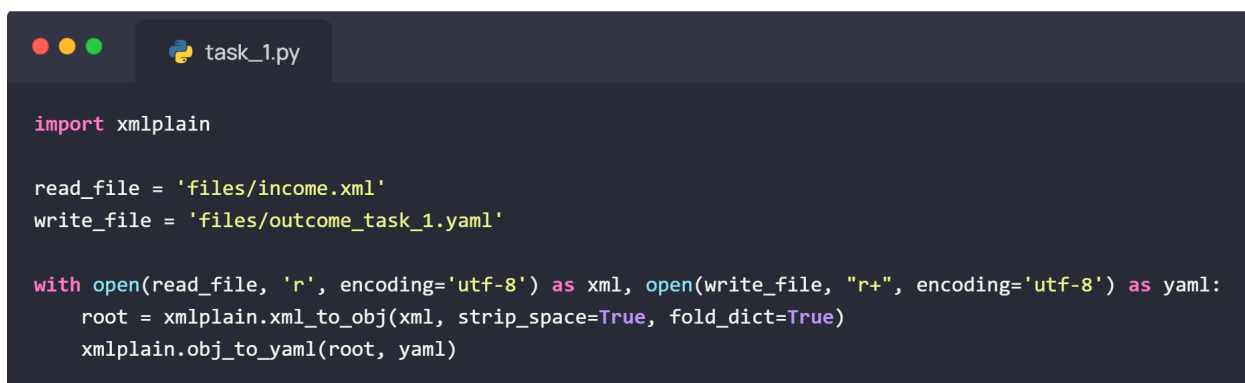


```
"расписание":
  "среда":
    "пара":
      -
        "предмет": "Информатика"
        "время_начала": "8:20"
        "время_окончания": "9:50"
        "вид_занятия": "Лекция"
        "преподаватель": "Балакшин Павел Валерьевич"
        "аудитория": 1216
      -
        "предмет": "Основы профессиональной деятельности"
        "время_начала": "10:00"
        "время_окончания": "11:30"
        "вид_занятия": "Лекция"
        "преподаватель": "Клименков Сергей Викторович"
        "аудитория": 1216
      -
        "предмет": "Бадминтон"
        "время_начала": "11:40"
        "время_окончания": "13:10"
        "вид_занятия": "Спорт"
        "преподаватель": "Трифонов Владислав Олегович"
        "аудитория": 4405
```

Рисунок 3 - Результат на YAML

Дополнительное задание №1

Для перевода файла из XML в YAML применим библиотеку `xmlplain`. Теперь код на Python код будет выглядеть как на Рисунке 4:



```
import xmlplain

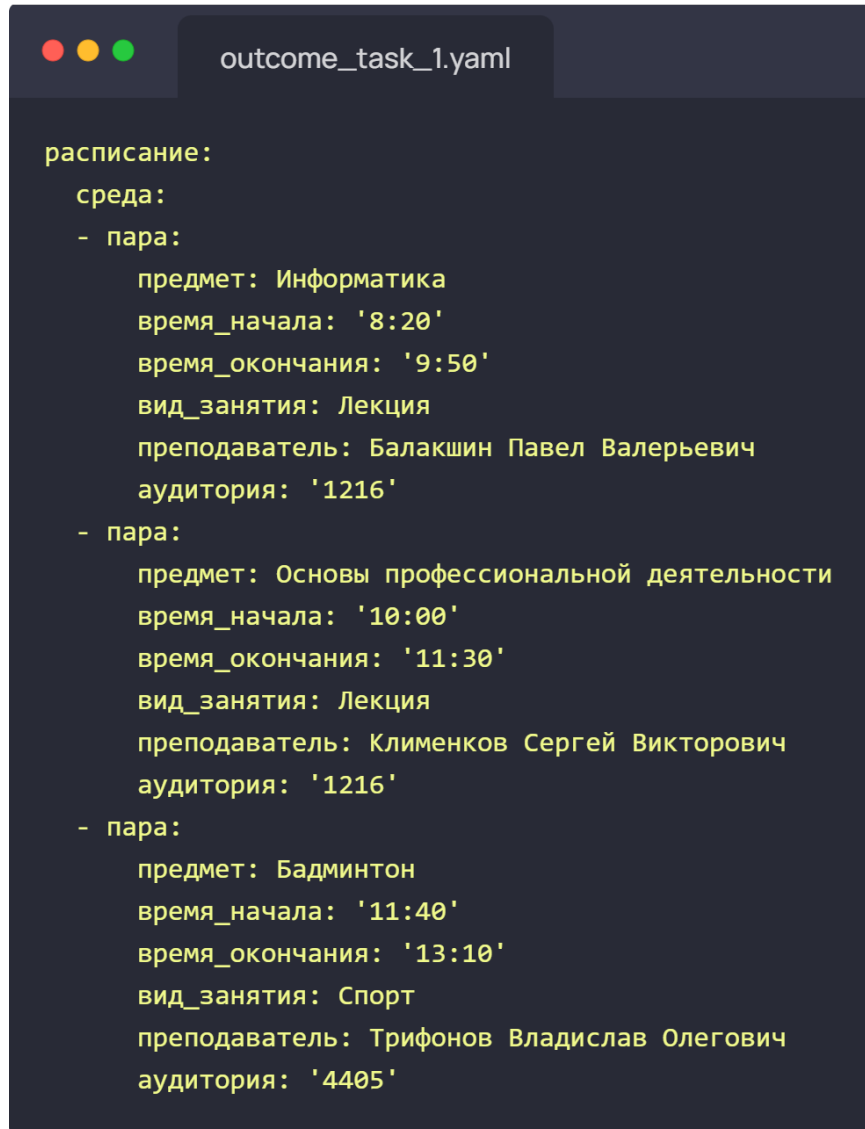
read_file = 'files/income.xml'
write_file = 'files/outcome_task_1.yaml'

with open(read_file, 'r', encoding='utf-8') as xml, open(write_file, "r+", encoding='utf-8') as yaml:
    root = xmlplain.xml_to_obj(xml, strip_space=True, fold_dict=True)
    xmlplain.obj_to_yaml(root, yaml)
```

Рисунок 4 - Код на языке Python с использованием библиотеки `xmlplain`

Сравнение полученных результатов

Полученный результат после работы кода из дополнительного задания №1 представлен на Рисунке 5. Он отличается от канонического оформления YAML документов (например, теги не взяты в кавычки), но при этом сохраняет свою функциональность.



```
outcome_task_1.yaml

расписание:
  среда:
    - пара:
        предмет: Информатика
        время_начала: '8:20'
        время_окончания: '9:50'
        вид_занятия: Лекция
        преподаватель: Балакшин Павел Валерьевич
        аудитория: '1216'
    - пара:
        предмет: Основы профессиональной деятельности
        время_начала: '10:00'
        время_окончания: '11:30'
        вид_занятия: Лекция
        преподаватель: Клименков Сергей Викторович
        аудитория: '1216'
    - пара:
        предмет: Бадминтон
        время_начала: '11:40'
        время_окончания: '13:10'
        вид_занятия: Спорт
        преподаватель: Трифонов Владислав Олегович
        аудитория: '4405'
```

Рисунок 5 - Результат дополнительного задания №1 в YAML

Дополнительное задание №2

Воспользуемся библиотекой `re`, чтобы использовать регулярные выражения и напишем код с их использованием, представленный на Рисунке 6:

```

task_2.py

import re

def main():
    read_file = "files/income.xml"
    write_file = "files/outcome_task_2.yaml"
    base(read_file, write_file)

def base(read_file, write_file):
    with open(read_file, mode='r', encoding='utf-8') as xml:
        text = xml.read()

    regex_1 = re.compile(r'>([^\<\d]+)</>')
    update_part_1 = regex_1.sub(r'>"1"</>', text)

    regex_2 = re.compile(r'</\w+>')
    update_part_2 = regex_2.sub(r'', update_part_1)

    regex_3 = re.compile(r'<(\w+)>')
    update_part_3 = regex_3.sub(r'"1": ', update_part_2)

    regex_4 = re.compile(r'<[?.*\?>')
    update_part_4 = regex_4.sub(r'', update_part_3)

    regex_5 = re.compile(r'(\d+:\d+)')
    update_part_5 = regex_5.sub(r'"1"', update_part_4)

    regex_6 = re.compile(r': (\d+)\\"')
    update_part_6 = regex_6.sub(r': 1', update_part_5)

    regex_7 = re.compile(r'\t\t\t\n')
    update_part_7 = regex_7.sub(r'\t\t\t-\n', update_part_6)

    regex_8 = re.compile(r'\n\t\t\t\w+":')
    inside = regex_8.finditer(update_part_7)

    if inside:
        delete = inside.__next__().end()
        update_part_7 = update_part_7[:delete] + regex_8.sub('', update_part_7[delete:])

    line = regex_8.search(update_part_7)
    if line:
        position = line.end()
        new_line = update_part_7[:position] + '\n\t\t\t\t-\n' + update_part_7[position:]

    result = new_line[1:618]

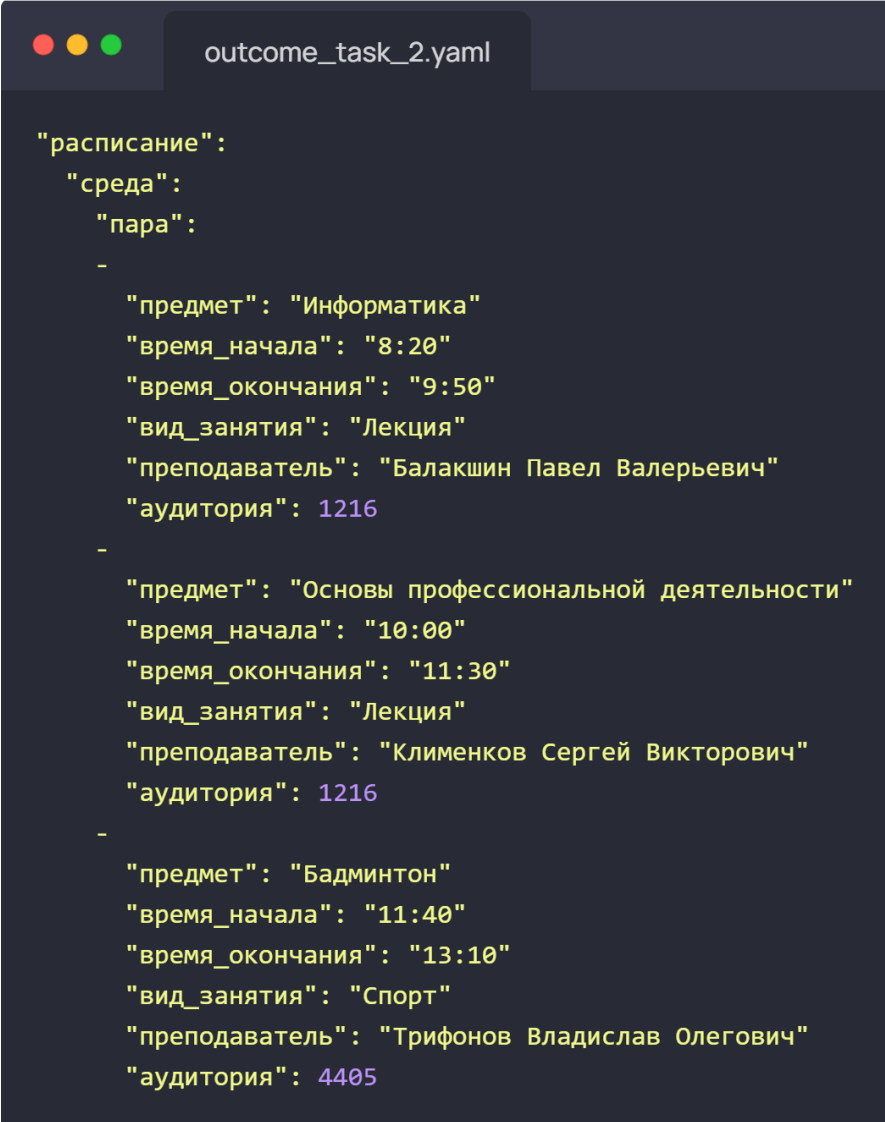
    with open(write_file, mode='w', encoding='utf-8') as yaml:
        yaml.write(result)

if __name__ == "__main__":
    main()

```

Рисунок 6 - Код с использованием регулярных выражений

В результате получается YAML файл, представленный на Рисунке 7.



```
"расписание":
  "среда":
    "пара":
      -
        "предмет": "Информатика"
        "время_начала": "8:20"
        "время_окончания": "9:50"
        "вид_занятия": "Лекция"
        "преподаватель": "Балакшин Павел Валерьевич"
        "аудитория": 1216
      -
        "предмет": "Основы профессиональной деятельности"
        "время_начала": "10:00"
        "время_окончания": "11:30"
        "вид_занятия": "Лекция"
        "преподаватель": "Клименков Сергей Викторович"
        "аудитория": 1216
      -
        "предмет": "Бадминтон"
        "время_начала": "11:40"
        "время_окончания": "13:10"
        "вид_занятия": "Спорт"
        "преподаватель": "Трифонов Владислав Олегович"
        "аудитория": 4405
```

Рисунок 7 - Результат дополнительного задания №2 в YAML

Сравнение полученных результатов

При использовании регулярных выражений результат аналогичен основному заданию, однако код, написанный с их применением, более универсален поскольку большинство регулярных выражений можно использовать при любых входных данных, написанных на языке XML.

Дополнительное задание №3

Дополним исходные данные, добавив ещё один учебный день. Теперь XML файл выглядит так, как представлено на Рисунке 8.

```
<?xml version="1.0" encoding="UTF-8"?>
<расписание>
  <среда>
    <пара>
      <предмет>Информатика</предмет>
      <время_начала>8:20</время_начала>
      <время_окончания>9:50</время_окончания>
      <вид_занятия>Лекция</вид_занятия>
      <преподаватель>Балакшин Павел Валерьевич</преподаватель>
      <аудитория>1216</аудитория>
    </пара>
    <пара>
      <предмет>Основы профессиональной деятельности</предмет>
      <время_начала>10:00</время_начала>
      <время_окончания>11:30</время_окончания>
      <вид_занятия>Лекция</вид_занятия>
      <преподаватель>Клименков Сергей Викторович</преподаватель>
      <аудитория>1216</аудитория>
    </пара>
    <пара>
      <предмет>Бадминтон</предмет>
      <время_начала>11:40</время_начала>
      <время_окончания>13:10</время_окончания>
      <вид_занятия>Спорт</вид_занятия>
      <преподаватель>Трифонов Владислав Олегович</преподаватель>
      <аудитория>4405</аудитория>
    </пара>
  </среда>
  <четверг>
    <пара>
      <предмет>История России и мира в XX веке.</предмет>
      <время_начала>10:00</время_начала>
      <время_окончания>11:30</время_окончания>
      <вид_занятия>Практика</вид_занятия>
      <преподаватель>Айба Тамара Гурамовна</преподаватель>
      <аудитория>1410</аудитория>
    </пара>
    <пара>
      <предмет>История России и мира в XX веке.</предмет>
      <время_начала>15:20</время_начала>
      <время_окончания>16:50</время_окончания>
      <вид_занятия>Лекция</вид_занятия>
      <преподаватель>Пригодич Никита Дмитриевич</преподаватель>
      <аудитория>1419</аудитория>
    </пара>
  </четверг>
</расписание>
```

Рисунок 8 - Исходные данные в XML для дополнительного задания №3

В качестве формальной грамматики используем словарь, где $\alpha \rightarrow \beta = \alpha: \beta$. Напишем программу на Python, представленную на Рисунке 9.

```
task_3.py

import re

def main():
    input_file_path = 'files/income_task_3.xml'
    output_file_path = 'files/outcome_task_3.yaml'
    process_xml(input_file_path, output_file_path)
    remove_empty_lines(output_file_path)

def process_xml(input_file, output_file):
    xml_name = re.compile(r'<?xml version="1\.\d" encoding="UTF-8"?>')
    close_tag_xml = re.compile(r'</[^\>\/]+>')
    string = re.compile(r'(<=>)([^\>\/]+)(?=<)')
    open_quote = re.compile(r'<')
    close_xml = re.compile(r'>')
    repeat_xml = re.compile(r'<нара>')

    yaml_name = ''
    close_tag_yaml = ''
    open_yaml = ''
    close_yaml = ''
    repeat_yaml = '-'

    grammar = {
        xml_name: yaml_name,
        repeat_xml: repeat_yaml,
        string: lambda match: f'"{match.group(1)}"',
        close_tag_xml: close_tag_yaml,
        open_quote: open_yaml,
        close_xml: close_yaml,
    }

    with open(input_file, 'r', encoding='utf-8') as infile, open(output_file, 'w', encoding='utf-8') as outfile:
        for line in infile:
            for pattern, replacement in grammar.items():
                line = re.sub(pattern, replacement, line)
            outfile.write(line)

def remove_empty_lines(output_file):
    with open(output_file, 'r', encoding='utf-8') as file:
        strings = file.readlines()

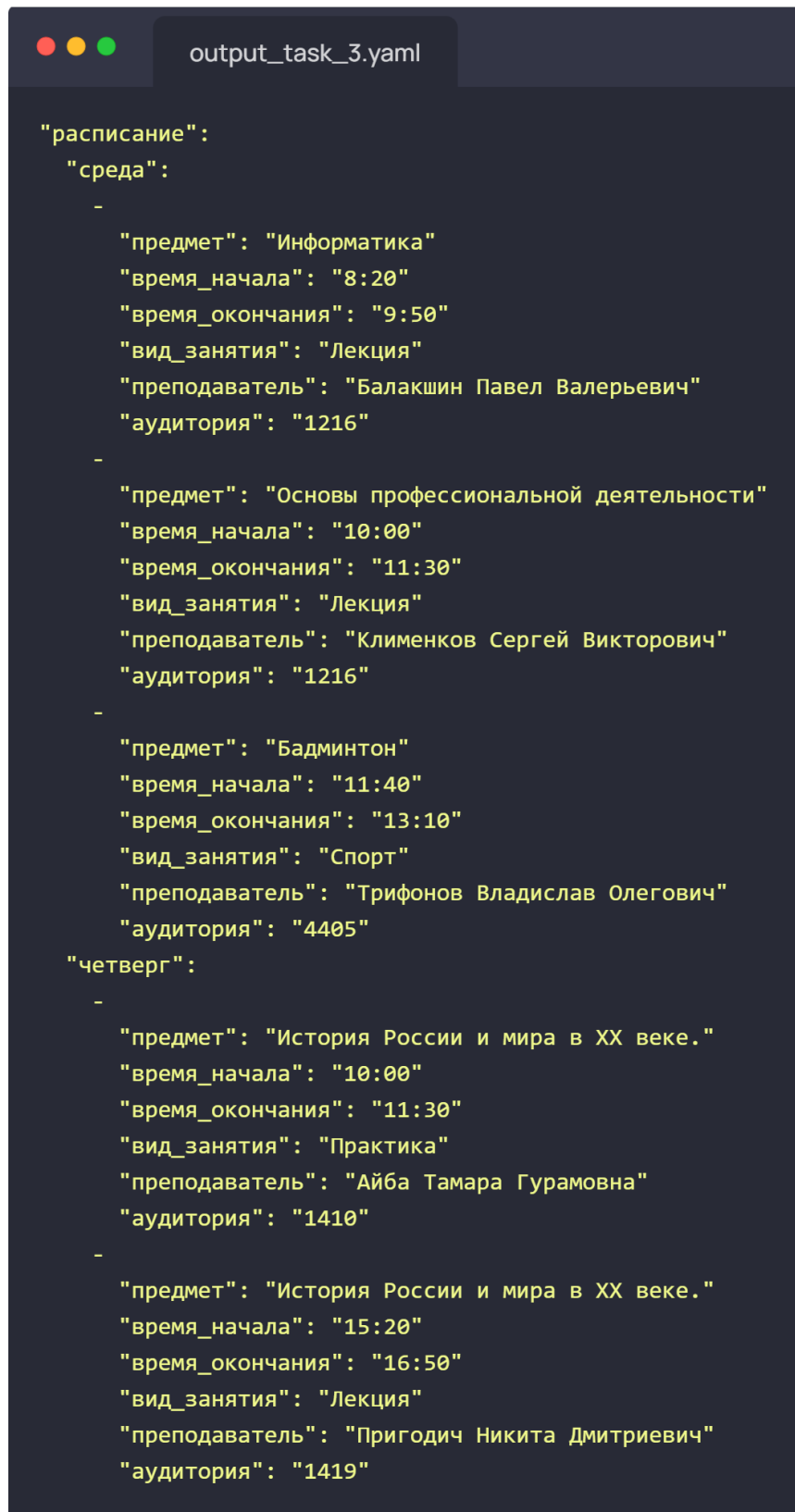
    non_empty = [strin for strin in strings if strin.strip() != '']

    with open(output_file, 'w', encoding='utf-8') as file:
        file.writelines(non_empty)

if __name__ == "__main__":
    main()
```

Рисунок 9 - Код для Дополнительного задания №3

В результате работы программы получим YAML файл, представленный на Рисунке 10.



```
"расписание":
  "среда":
    -
      "предмет": "Информатика"
      "время_начала": "8:20"
      "время_окончания": "9:50"
      "вид_занятия": "Лекция"
      "преподаватель": "Балакшин Павел Валерьевич"
      "аудитория": "1216"
    -
      "предмет": "Основы профессиональной деятельности"
      "время_начала": "10:00"
      "время_окончания": "11:30"
      "вид_занятия": "Лекция"
      "преподаватель": "Клименков Сергей Викторович"
      "аудитория": "1216"
    -
      "предмет": "Бадминтон"
      "время_начала": "11:40"
      "время_окончания": "13:10"
      "вид_занятия": "Спорт"
      "преподаватель": "Трифонов Владислав Олегович"
      "аудитория": "4405"
  "четверг":
    -
      "предмет": "История России и мира в XX веке."
      "время_начала": "10:00"
      "время_окончания": "11:30"
      "вид_занятия": "Практика"
      "преподаватель": "Айба Тамара Гурамовна"
      "аудитория": "1410"
    -
      "предмет": "История России и мира в XX веке."
      "время_начала": "15:20"
      "время_окончания": "16:50"
      "вид_занятия": "Лекция"
      "преподаватель": "Пригодич Никита Дмитриевич"
      "аудитория": "1419"
```

Рисунок 10 - Результат дополнительного задания №3 в YAML

Сравнение полученных результатов

Используя формальную грамматику, мы получили такой же результат выходного файла, однако такой подход помогает обрабатывать файлы с большим объёмом данных.

Дополнительное задание №4

Для увеличения времени исполнения в 100 и в 1000 раз дополним исходный код программы так, как представлено на Рисунке 11.

```
task_1.py

import time

start_time = time.time()

for i in range(1000):
    read_file = 'files/income.xml'
    write_file = 'files/outcome_task_4.yaml'

    with open(read_file, mode='r', encoding='utf-8') as xml, open(write_file, mode='w', encoding='utf-8') as yaml:

        para_cnt = 0
        for line in xml:
            start = line.find("</")
            end = line.find(">", start)
            while start != -1 and end != -1:
                line = line[start] + line[end + 1:]

                start = line.find("</")
                end = line.find(">", start)
            line = line.replace('<', '')
            line = line.replace('>', '')

            if "napa" in line:
                para_cnt += 1

            if "napa" in line and para_cnt > 1:
                line = line.replace("napa:", "-")

            if "лекция" in line:
                line = line.replace("лекция", "Лекция")

            if "Информатика" in line:
                line = line.replace("информатика", "Информатика")

            if "Основы профессиональной деятельности" in line:
                line = line.replace("Основы профессиональной деятельности", "Основы профессиональной деятельности")

            if "Бадминтон" in line:
                line = line.replace("бадминтон", "Бадминтон")

            if "Спорт" in line:
                line = line.replace("спорт", "Спорт")

            if "Балахнин Павел Валерьевич" in line:
                line = line.replace("Балахнин Павел Валерьевич", "Балахнин Павел Валерьевич")

            if "Клименков Сергей Викторович" in line:
                line = line.replace("Клименков Сергей Викторович", "Клименков Сергей Викторович")

            if "Трифонов Владислав Олегович" in line:
                line = line.replace("Трифонов Владислав Олегович", "Трифонов Владислав Олегович")

            if "8:20" in line:
                line = line.replace("8:20", "8:20")

            if "9:50" in line:
                line = line.replace("9:50", "9:50")

            if "10:00" in line:
                line = line.replace("10:00", "10:00")

            if "11:30" in line:
                line = line.replace("11:30", "11:30")

            if "11:40" in line:
                line = line.replace("11:40", "11:40")

            if "13:10" in line:
                line = line.replace("13:10", "13:10")

            if "?xml version='1.0' encoding='UTF-8'?" not in line:
                yaml.write(line)

        with open(write_file, 'r', encoding='utf-8') as file:
            strings = file.readlines()

            strings.insert(3, " " + '\n')
            non_empty = [strin for strin in strings if strin.strip() != '']

        with open(write_file, 'w', encoding='utf-8') as file:
            file.writelines(non_empty)

end_time = time.time()

elapsed_time = end_time - start_time

print(f"Время выполнения программы: {elapsed_time} секунд")
```

Рисунок 11 - Код для Дополнительного задания №4

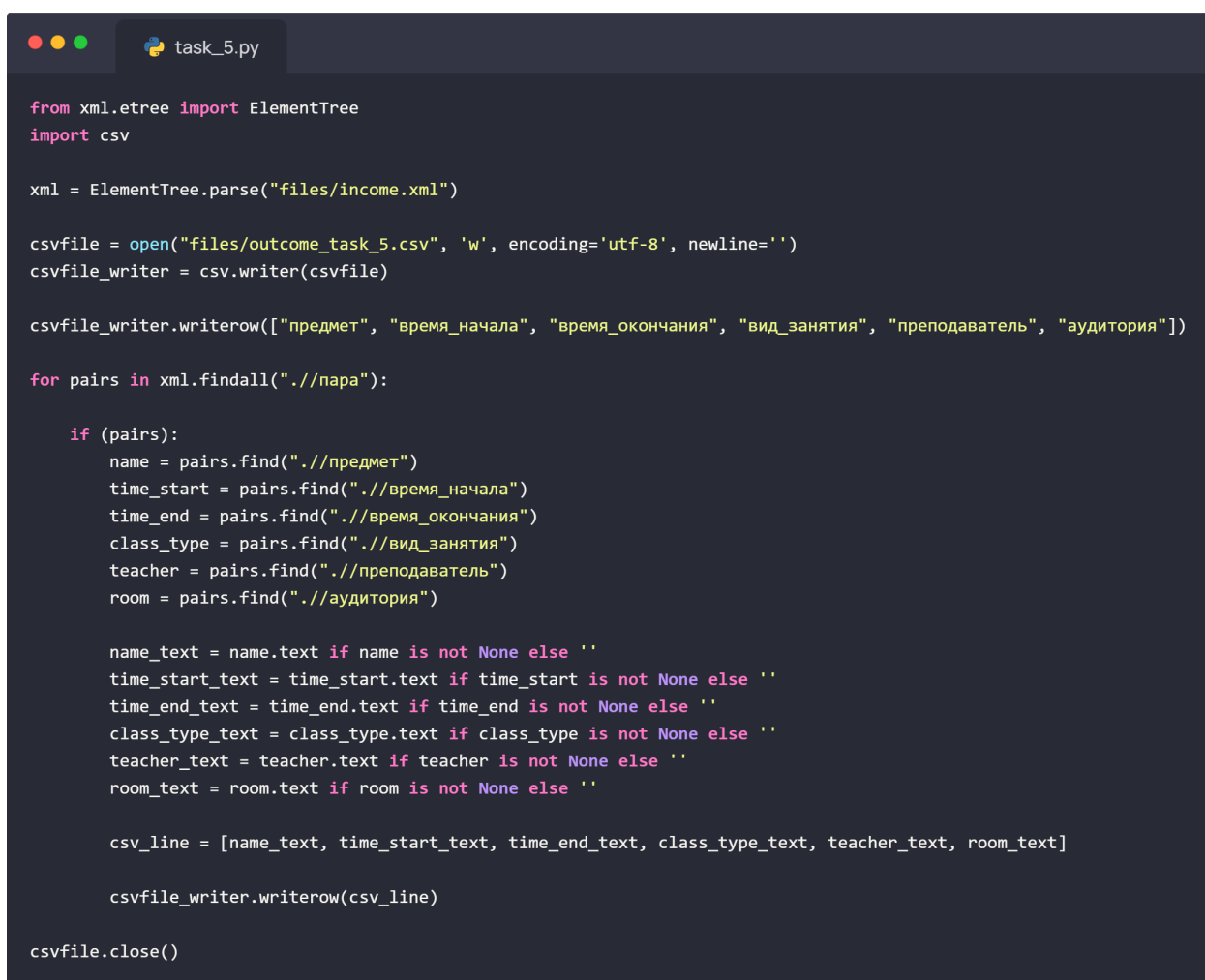
Сравнение полученных результатов

Таблица 1 - Сравнение времени выполнения программы

Выполнение	Однократное	Стократное	Тысячекратное
Время (в секундах)	0,013997316360473633	0,43395090103149414	2,53118896484375

Дополнительное задание №5

Перепишем исходный код программы, чтобы он осуществлял парсинг и конвертацию исходного файла из XML в CSV. Необходимый код представлен на Рисунке 12.



```
task_5.py

from xml.etree import ElementTree
import csv

xml = ElementTree.parse("files/income.xml")

csvfile = open("files/outcome_task_5.csv", 'w', encoding='utf-8', newline='')
csvfile_writer = csv.writer(csvfile)

csvfile_writer.writerow(["предмет", "время_начала", "время_окончания", "вид_занятия", "преподаватель", "аудитория"])

for pairs in xml.findall("./пара"):

    if (pairs):
        name = pairs.find("./предмет")
        time_start = pairs.find("./время_начала")
        time_end = pairs.find("./время_окончания")
        class_type = pairs.find("./вид_занятия")
        teacher = pairs.find("./преподаватель")
        room = pairs.find("./аудитория")

        name_text = name.text if name is not None else ''
        time_start_text = time_start.text if time_start is not None else ''
        time_end_text = time_end.text if time_end is not None else ''
        class_type_text = class_type.text if class_type is not None else ''
        teacher_text = teacher.text if teacher is not None else ''
        room_text = room.text if room is not None else ''

        csv_line = [name_text, time_start_text, time_end_text, class_type_text, teacher_text, room_text]

        csvfile_writer.writerow(csv_line)

csvfile.close()
```

Рисунок 12 - Код для дополнительного задания №5

В результате работы программы получаем CSV файл, представленный на Рисунке 13.

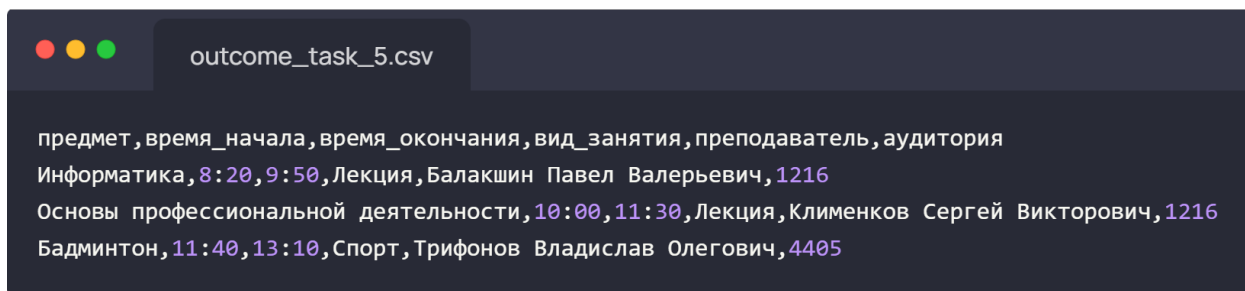


Рисунок 13 - Результат дополнительного задания №5 в CSV

Анализ полученных результатов

В результате работы программы был получен файл в формате CSV с разделением по запятой, который можно использовать в офисных пакетах, чтобы составить электронную таблицу. Пример такой таблицы представлен на Рисунке 14.

предмет	время_начала	время_окончания	вид_занятия	преподаватель	аудитория
Информатика	8:20	9:50	Лекция	Балакшин Павел Валерьевич	1216
Основы профессиональной деятельности	10:00	11:30	Лекция	Клименков Сергей Викторович	1216
Бадминтон	11:40	13:10	Спорт	Трифонов Владислав Олегович	4405

Рисунок 14 - CSV файл в виде таблицы

Заключение

В ходе выполнения данной лабораторной работы я познакомился с различными языками разметки, научился осуществлять парсинг и конвертацию одного файла в другой, используя регулярные выражения, формальную грамматику и встроенные библиотеки.

Список литературы

Волкова И. А. Вылиток А. А., Руденко Т. В. Формальные грамматики и языки. Элементы теории трансляции [Книга]. - Москва : Факультет вычислительной математики и кибернетики МГУ им. М. В. Ломоносов, 2009. - 3-е : стр. 115.

Джеффри Фридл Регулярные выражения [Книга] / ред. А. Галунов / перев. Матвеева Е. Киселева А.. - Санкт-Петербург - Москва : СИМВОЛ, 2008. - 3-е : стр. 598. - 9785932861219.

Лопес Феликс Ромеро Виктор Освоение регулярных выражений Python [Книга]. - Бирмингем : Packt Publishing, 2014. - стр. 110. - 9781783283156.