Федеральное государственное автономное образовательное учреждение высшего образования
«Санкт-Петербургский национальный исследовательский университет **информационных технологий, механики и оптики»**

Факультет ПИИКТ

Дисциплина: Информатика

Лабораторная работа №4
«Исследование языков разметки документов»

Вариант №5

Выполнил: Братчиков Иван Станиславович
Группа: P3101

2018г

Цель задания:
Овладеть знаниями о различных современных языках разметки документов и форматах данных, навыками обработки данных с помощью языка Python 3.X.
~~~~~~
Задание:

1. Изучить форму Бэкуса – Наура
2. Изучить особенности языков разметки/форматов JSON, YAML, XML, PROTOBUF
3. Понять устройство страницы с расписанием для своей группы
4. Исходя из структуры расписания конкретного дня сформировать файл с расписанием в формате, указанном в задании в качестве исходного
5. Написать программу на языка Python, которая бы осуществляла парсинг и конвертацию исходного файла в новый. Нельзя использовать готовые библиотеки, кроме re.
6. Номер варианта определить как остаток деления на 21 порядкового номера в списке группы в ISU
7. Написать вывод по итогам выполнения лабораторной работы
8. Проверить, что все пункты задания выполнены и выполнены верно

<u>5 вариант – исходный формат JSON, результирующий формат XML</u>
~~~~~~
Решение:

sync.py:

```
#!/usr/bin/python3
import sys

# built-in regexp lib
import re
import requests
from bs4 import BeautifulSoup

SCHEDULE_URL = 'http://www.ifmo.ru/ru/schedule/0/{}/raspisanie_zanyatiy.htm'

def parse_schedule(group):
   response = requests.get(SCHEDULE_URL.format(group))  # .format() fill placeholders in string
with variables contents
   html = response.text

   bs = BeautifulSoup(html, features='lxml')  # import parsing lib with lxml library for faster parsing
   content_block = bs.find('article', class_='content_block')

    if 'Расписание не найдено' in str(content_block):
      print(f'Cannot parse group {group} - schedule not found!')
      return

    rasp_tabl_days = content_block.find_all('div', class_='rasp_tabl_day')

   s = '{"Monday": {'

    i = 0

   classes = rasp_tabl_days[0].find('table').find_all('tr')[:-1]
```

```
    for row in classes:
        if i != (len(classes) - 1):
                    s = s + '"class ' + str(i) + '": ' + '{"lesson": ' +  "" + str(row.find('td',
         class_='lesson').find('dd').text) +  "" + "," + '"time": ' +  "" + str(row.find('td',
         class_='time').find('span').text) +  "" +"," + '"room": ' +   "" + str(row.find('td',
         class_='room').find('dd').text) +  "" + "}, "
                else: s = s + '"class ' + str(i) + '": ' + '{"lesson": ' +  "" + str(row.find('td',
         class_='lesson').find('dd').text) +  "" + "," + '"time": ' +  "" + str(row.find('td',
         class_='time').find('span').text) +  "" +"," + '"room": ' +   "" + str(row.find('td',
         class_='room').find('dd').text) +  "" + "}}}"
        i += 1

    f = open('data.txt', "w")
    f.write(s)
    f.close

if __name__ == "__main__":
    # Called as program, not just imported

    # Check if we called with arguments
    if len(sys.argv) < 2:
        print('Usage: sync.py <group 1> <group 2> ... <group N>')

    # sys.argv[1:] means that we took a _slice_ from sys.argv - from element with index 1 to end of
array
    for group in sys.argv[1:]:
        result = re.match(r'\w\d{4}', group)
        if not result:
            print(f'Cannot parse group {group}, maybe you make mistake?')
            sys.exit(0)

    # If all groups are correct, we can start parsing
    for group in sys.argv[1:]:
        parse_schedule(group)
```

json2xml.py:

```
def json2xml(json_obj, line = " "):
    result = []
    json_obj_type = type(json_obj)

    if json_obj_type is list:
        for sub_elem in json_obj:
            result.append(json2xml(sub_elem, line))

        return "\n".join(result)
```

```
    if json_obj_type is dict:
        for tag in json_obj:
            sub_obj = json_obj[tag]
            result.append("%s<%s>" % (line, tag))
            result.append(json2xml(sub_obj, "\t" + line))
            result.append("%s</%s>" % (line, tag))

        return "\n".join(result)

    return "%s%s" % (line, json_obj)

with open("data.txt","r") as f:
    s = f.readlines()
    print(print('\n'+ "JSON:" + '\n'), eval(s[0]))
    f.close()

k = s[0]
d = eval(k)
print('\n'*2 + "XML:")
print(json2xml(d))
```

~~~~~~

Вывод: В ходе выполнения лабораторной работы я узнал о различным языках разметки/форматов и написал простой конвертер формата json в xml.
~~~~~~