

Университет ИТМО

Информатика

Отчет по лабораторной работе №4 (Исследование протоколов, форматов обмена информацией и языков разметки документов)

ФИО студента: Готовко Алексей Владимирович

Номер варианта: 13533 (номер ИСУ: 335151)

Направление подготовки: 09.03.04 (СППО)

Учебная группа: Р3119

ФИО преподавателя: Балакшин Павел Валерьевич

Санкт-Петербург, 2021 г.

Оглавление

1.	Задание	3
2.	Выполнение работы	4
3.	Вывод	8

1. Задание

- 1.1 Написать программу на языке Python 3.x, которая бы осуществляла парсинг и конвертацию исходного файла в новый.
- 1.2 Найти готовые библиотеки, осуществляющие аналогичный парсинг и конвертацию файлов. Переписать исходный код, применив найденные библиотеки. Регулярные выражения также нельзя использовать.
- 1.3 Переписать исходный код, добавив в него использование регулярных выражений.
- 1.4 Используя свою исходную программу из обязательного задания, программу из дополнительного задания №1 и программу из дополнительного задания №2, сравнить десятикратное время выполнения парсинга + конвертации в цикле.
- 1.5 Переписать исходную, чтобы она осуществляла парсинг и конвертацию исходного файла в любой другой формат (кроме JSON, YAML, XML, HTML): PROTOBUF, TSV, CSV, WML и т.п.

Вариант задания:

JSON – YAML – CSV

2. Выполнение работы

```
1 # json2yaml converter (no libs, no regexes)
2 import time
3
4 def main():
5     out = ["---\n"]
6     flagNewBlock = False
7     with open("timetable.json", 'r', encoding="utf-8") as f_in:
8         line = f_in.readline()
9         while line:
10             # format tabulation and dashes
11             if flagNewBlock:
12                 flagNewBlock = False
13                 tabs = line.count('\t')
14                 line = line.replace('\t', '')
15                 spaces = ' ' * (tabs - 2)
16                 line = spaces + '- ' + line
17             else:
18                 line = line.replace('\t', ' ')
19                 if '{' in line:
20                     flagNewBlock = True
21             # format symbols
22             for c in "\\\",{}[]":
23                 line = line.replace(c, '')
24             # format colons
25             line = line.split(' ')
26             for idx, word in enumerate(line):
27                 if len(word) < 3:
28                     continue
29                 flagNewLine, flagColon = False, False
30                 if word[-1] == '\n':
31                     flagNewLine = True
32                     word = word[:-1]
33                 if word[-1] == ':':
34                     flagColon = True
35                     word = word[:-1]
36                 if ':' in word:
37                     word = '\\' + word + '\\'
38                 if flagColon:
39                     word += ':'
40                 if flagNewLine:
41                     word += '\n'
42                 line[idx] = word
43             line = ' '.join(line)
44
45             out.append(line)
46             line = f_in.readline()
47
48     for line in out:
49         s = line.split()
50         if not s:
51             out.remove(line)
52
53     with open("timetable.yaml", 'w', encoding="utf-8") as f_out:
54         f_out.write(''.join(out))
55
56 # json2yaml (with libs, no regexes)
57 import json, yaml
58
59 def main():
60     with open("timetable.json", 'r', encoding="utf-8") as jsonFile:
61         jsonData = json.load(jsonFile)
62     with open("task1.yaml", 'w', encoding="utf-8") as yamlFile:
63         yamlData = yaml.dump(jsonData, allow_unicode=True)
64         yamlFile.write(yamlData)
```

```

1 # json2yaml converter (no libs, with regexes)
2 import re
3
4 def main():
5     out = ["---\n"]
6     flagNewBlock = False
7     with open("timetable.json", 'r', encoding="utf-8") as f_in:
8         line = f_in.readline()
9         while line:
10             # format tabulation and dashes
11             if flagNewBlock:
12                 flagNewBlock = False
13                 line = re.sub(r'\t+', ' ' * (line.count('\t') - 2) + '- ', line)
14             else:
15                 line = line.replace('\t', ' ')
16                 if '{' in line:
17                     flagNewBlock = True
18             # format symbols
19             line = re.sub(r'[\\"', {}\[ \]]', '', line)
20             # format colons
21             line = line.split(' ')
22             for idx, word in enumerate(line):
23                 if len(word) < 3:
24                     continue
25                 flagNewLine, flagColon = False, False
26                 if word[-1] == '\n':
27                     flagNewLine = True
28                 word = word[:-1]
29                 if word[-1] == ':':
30                     flagColon = True
31                 word = word[:-1]
32                 if ':' in word:
33                     word = '\\' + word + '\\'
34                 if flagColon:
35                     word += ':'
36                 if flagNewLine:
37                     word += '\n'
38                 line[idx] = word
39             line = ' '.join(line)
40
41             out.append(line)
42             line = f_in.readline()
43
44         for line in out:
45             s = re.sub(r'\s+', '', line)
46             if not s:
47                 out.remove(line)
48
49         with open("task2.yaml", 'w', encoding="utf-8") as f_out:
50             f_out.write(''.join(out))

```

Проверка времени исполнения:

```
1 import time
2 import json2yaml_no_lib
3 import json2yaml_regex
4 import json2yaml_lib
5
6
7 start = time.process_time()
8 for i in range(100):
9     open("timetable.yaml", 'w').close()
10    json2yaml_no_lib.main()
11 print(f'No libs, no regex: {time.process_time() - start}')
12
13 start = time.process_time()
14 for i in range(100):
15     open("task1.yaml", 'w').close()
16     json2yaml_regex.main()
17 print(f'No libs, with regex: {time.process_time() - start}')
18
19 start = time.process_time()
20 for i in range(100):
21     open("task2.yaml", 'w').close()
22     json2yaml_lib.main()
23 print(f'With lib: {time.process_time() - start}')
24
```

Было использовано стократное, а не десятикратное, время выполнения для большей наглядности результата

```
No libs, no regex: 0.078125
No libs, with regex: 0.0625
With lib: 0.1875
```

Исходный файл (JSON)

```
1 ---
2 - Monday:
3   - discipline: Дискретная математика (практика)
4     lecturer: Поляков Владимир Иванович
5     start_time: '11:40'
6     end_time: '13:10'
7     place: Кронверкский пр. д.49 лит.А
8     lecture_room: 369А
9     week_parity: четная неделя
10    studying_format: Очно - дистанционный
11  - discipline: Дискретная математика (практика)
12    lecturer: Поляков Владимир Иванович
13    start_time: '13:30'
14    end_time: '15:00'
15    place: Кронверкский пр. д.49 лит.А
16    lecture_room: 369А
17    week_parity: четная неделя
18    studying_format: Очно - дистанционный
19
20
21
```

Полученные файлы (YAML и CSV):

```
1 {
2   "Monday": [
3     {
4       "discipline": "Дискретная математика (практика)",
5       "lecturer": "Поляков Владимир Иванович",
6       "start_time": "11:40",
7       "end_time": "13:10",
8       "place": "Кронверкский пр., д.49, лит.А",
9       "lecture_room": "369А",
10      "week_parity": "четная неделя",
11      "studying_format": "Очно - дистанционный"
12    },
13
14    {
15      "discipline": "Дискретная математика (практика)",
16      "lecturer": "Поляков Владимир Иванович",
17      "start_time": "13:30",
18      "end_time": "15:00",
19      "place": "Кронверкский пр., д.49, лит.А",
20      "lecture_room": "369А",
21      "week_parity": "четная неделя",
22      "studying_format": "Очно - дистанционный"
23    }
24  ]
25 }
```

```
1 ,Monday
2
3 0,{"discipline": 'Дискретная математика (практика)', 'lecturer': 'Поляков Владимир
Иванович', 'start_time': '11:40', 'end_time': '13:10', 'place': 'Кронверкский пр.,
д.49, лит.А', 'lecture_room': '369А', 'week_parity': 'четная неделя',
'studying_format': 'Очно - дистанционный'}"
4
5 1,{"discipline": 'Дискретная математика (практика)', 'lecturer': 'Поляков Владимир
Иванович', 'start_time': '13:30', 'end_time': '15:00', 'place': 'Кронверкский пр.,
д.49, лит.А', 'lecture_room': '369А', 'week_parity': 'четная неделя',
'studying_format': 'Очно - дистанционный'}"
6
7
```

3. Вывод

В процессе выполнения лабораторной работы удалось познакомиться с протоколами, форматами обмена информацией и языками разметки; реализовать парсер-конвертер между форматами JSON и YAML.

Список литературы

- [Презентация лекции №4](https://isu.ifmo.ru/pls/apex/f?p=2002:0:121542816407093::DWNLD_F:NO::FILE:BC31BC6D79279D03C961166D5D910478)
(https://isu.ifmo.ru/pls/apex/f?p=2002:0:121542816407093::DWNLD_F:NO::FILE:BC31BC6D79279D03C961166D5D910478)
- [Черновик методического пособия «Информатика»](https://vk.com/doc-31201840_566998093)
(https://vk.com/doc-31201840_566998093)