

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Дисциплина информатика

Лабораторная работа № 3

Выполнил студент

Суджян Эдуард Эдуардович

Группа № Р3121

Преподаватель: Болдырева Елена Александровна

Санкт-Петербург

2023

Задание 1 вариант: 0, 1, 5

Задание 2 вариант: 1

Задание 3 вариант: 1

Весь код доступен по этой ссылке

Оглавление

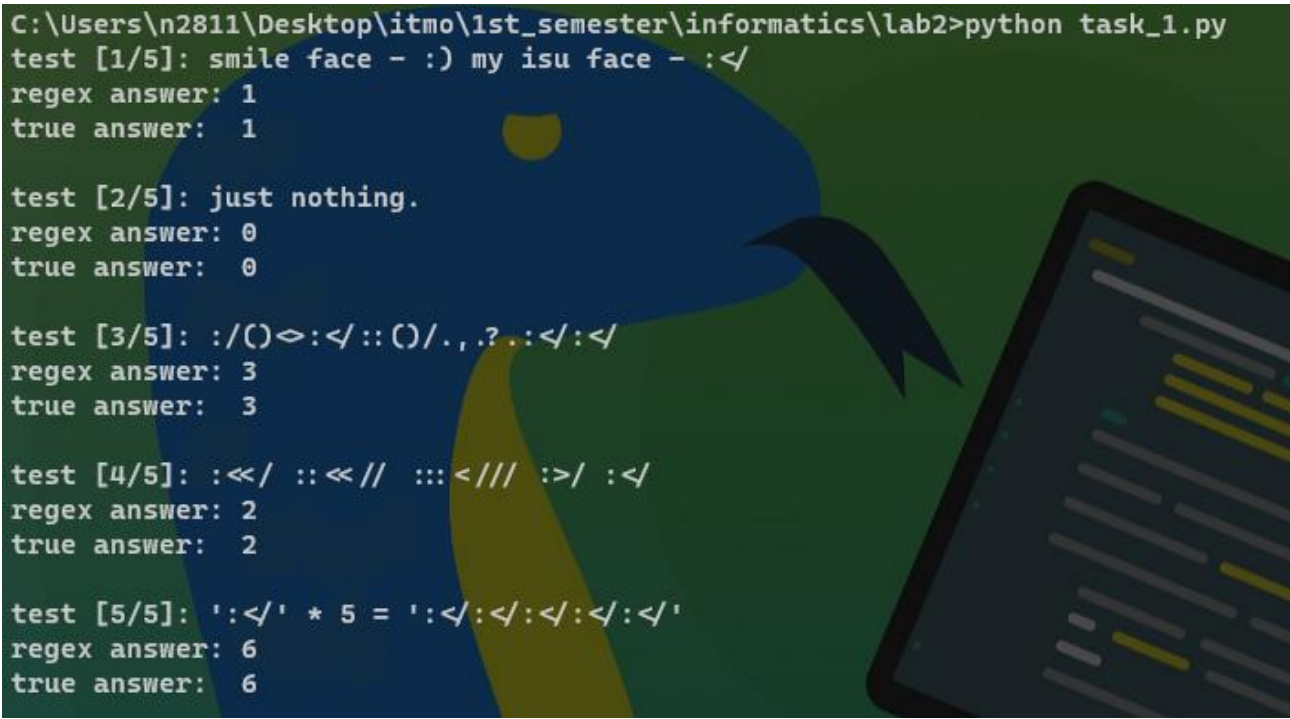
Задание 1.....	3
Решение.....	3
Задание 2.....	4
Решение.....	5
Задание 3.....	6
Решение.....	7
Дополнительное задание №1	11
Решение.....	11
Вывод	12
Список литературы.....	12

Задание 1

1) Реализуйте программный продукт на языке Python, используя регулярные выражения по варианту, представленному в таблице. 2) Для своей программы придумайте минимум 5 тестов. Каждый тест является отдельной сущностью, передаваемой регулярному выражению для обработки. Для каждого теста необходимо самостоятельно (без использования регулярных выражений) найти правильный ответ. После чего сравнить ответ, выданный программой, и полученный самостоятельно. 3) Программа должна считать количество смайликов определённого вида (вид смайлика описан в таблице вариантов) в предложенном тексте

Решение

```
1  import re
2
3
4  # :</
5
6  tests = [
7      "smile face - :) my isu face - :</",
8      "just nothing.",
9      ":(())<:</::()/. ,.?:</:</",
10     ";<</ ::<</// ::<</// :>/ :</",
11     "' :</' * 5 = ' :</:</:</:</:</'"
12 ]
13
14 answers = [
15     1,
16     0,
17     3,
18     2,
19     6
20 ]
21
22 pattern = r":</"
23
24 if __name__ == "__main__":
25     for i in range(len(tests)):
26         count = len(re.findall(pattern, tests[i]))
27
28         print(f"test [{i+1}/{len(tests)}]: {tests[i]}\nregex answer: {count}\ntrue answer: {answers[i]}\n")
29
```

The background of the terminal window features a large, stylized Python logo in blue and yellow. To the right, there is a dark laptop screen displaying lines of code in a light color.

```
C:\Users\n2811\Desktop\itmo\1st_semester\informatics\lab2>python task_1.py
test [1/5]: smile face - :) my isu face - :</
regex answer: 1
true answer: 1

test [2/5]: just nothing.
regex answer: 0
true answer: 0

test [3/5]: :/(O>:</::C)/.,.? :</:</
regex answer: 3
true answer: 3

test [4/5]: :<</ ::<<// :::</// :>/ :</
regex answer: 2
true answer: 2

test [5/5]: ':</' * 5 = ':<:<:<:<:<:'
regex answer: 6
true answer: 6
```

Задание 2

1) Реализуйте программный продукт на языке Python, используя регулярные выражения по варианту, представленному в таблице. 2) Для своей программы придумайте минимум 5 тестов. 3) Протестируйте свою программу на этих тестах.

Студент Вася очень любит курс «Компьютерная безопасность». Однажды Васе задали домашнее задание зашифровать данные, переданные в сообщении. Недолго думая, Вася решил заменить все целые числа на функцию от этого числа. Функцию он придумал не сложную $3x^2+5$, где x – исходное число. Помогите Васе с его домашним заданием.

Решение

```
1 import re
2
3
4 vasya_function = lambda x: 3 * x**2 + 5
5
6 task_test = "20 + 22 = 42"
7 task_answer = "1205 + 1457 = 5297"
8
9 my_tests = [
10     "123 + 321 - 456 = 789",
11     "-1---123---4-",
12     "nothing to correct",
13     "2 + 2 = 4",
14     "100101010100101",
15     "1 0 1 0 1 1 1 0 1 0 0 1 0"
16 ]
17
18 my_answers = [
19     "45392 + 309128 - 623813 = 1867568",
20     "-8---45392---53-",
21     "nothing to correct",
22     "17 + 17 = 53",
23     "30060636669181567242120630608",
24     "8 5 8 5 8 8 8 5 8 5 5 8 5"
25 ]
26
27 pattern = r"\d+"
28
29 if __name__ == "__main__":
30     task_numbers = list(map(int, re.findall(pattern, task_test)))
31     numbers_after_vasya = list(map(vasya_function, task_numbers))
32     result = re.sub(pattern, "{}", task_test).format(*numbers_after_vasya)
33
34     print(f"task test: {task_test}\nregex res: {result}\ntask answer: {task_answer}\n\n")
35
36     print("----- My tests -----")
37     for i in range(len(my_tests)):
38         task_numbers = list(map(int, re.findall(pattern, my_tests[i])))
39         numbers_after_vasya = list(map(vasya_function, task_numbers))
40         result = re.sub(pattern, "{}", my_tests[i]).format(*numbers_after_vasya)
41         print(f"test [{i+1}/{len(my_tests)}]: {my_tests[i]}\nregex answer: {result}\ntrue answer: {my_answers[i]}\n")
42
```

```
C:\Users\n2811\Desktop\itmo\1st_semester\informatics\lab2>python task_2.py
task test: 20 + 22 = 42
regex res: 1205 + 1457 = 5297
task answer: 1205 + 1457 = 5297

----- My tests -----
test [1/6]: 123 + 321 - 456 = 789
regex answer: 45392 + 309128 - 623813 = 1867568
true answer: 45392 + 309128 - 623813 = 1867568

test [2/6]: -1---123---4-
regex answer: -8---45392---53-
true answer: -8---45392---53-

test [3/6]: nothing to correct
regex answer: nothing to correct
true answer: nothing to correct

test [4/6]: 2 + 2 = 4
regex answer: 17 + 17 = 53
true answer: 17 + 17 = 53

test [5/6]: 100101010100101
regex answer: 30060636669181567242120630608
true answer: 30060636669181567242120630608

test [6/6]: 1 0 1 0 1 1 1 0 1 0 0 1 0
regex answer: 8 5 8 5 8 8 8 5 8 5 5 8 5
true answer: 8 5 8 5 8 8 8 5 8 5 5 8 5
```

Задание 3

1. Определить номер варианта как остаток деления номера в ИСУ на 36. В случае, если в данный день недели нет занятий, то увеличить номер варианта на восемь. 2. Изучить форму Бэкуса-Наура. 3. Изучить особенности протоколов и форматов обмена информацией между системами: JSON, YAML, XML. 4. Понять устройство страницы с расписанием для своей группы: <https://itmo.ru/ru/schedule/0/P3110/schedule.htm> 5. Исходя из структуры расписания конкретного дня, сформировать файл с расписанием в формате, указанном в задании в качестве исходного. 6. Обязательное задание: написать программу на языке Python 3.x, которая бы осуществляла парсинг и конвертацию исходного файла в новый. 7. Нельзя использовать готовые библиотеки, в том числе регулярные выражения в Python и библиотеки для загрузки XML-файлов. Json -> xml

Решение

```
1  {
2    "even week": [
3      {
4        "name": "LIFE SAFETY (LEK): INTRODUCTORY LECTURE",
5        "teacher": "Rusanov Dmitriy Yurevich",
6        "type": "distant",
7        "day": "monday",
8        "start_time": "18:40",
9        "end_time": "20:10"
10     }
11  ],
12
13  "odd week": [
14    {
15      "name": "LIFE SAFETY (LEK): INTRODUCTORY LECTURE",
16      "teacher": "Rusanov Dmitriy Yurevich",
17      "type": "distant",
18      "day": "monday",
19      "start_time": "18:40",
20      "end_time": "20:10"
21    },
22
23    {
24      "name": "LIFE SAFETY (LEK): INTRODUCTORY LECTURE",
25      "teacher": "Rusanov Dmitriy Yurevich",
26      "type": "distant",
27      "day": "monday",
28      "start_time": "18:40",
29      "end_time": "20:10"
30    }
31  ]
32 }
```

```
1  from typing import TextIO, NoReturn
2
3
4  F1_NAME = "schedule.json"
5  F2_NAME = "schedule.xml"
6
7
8  def delete_trash(data):
9      start_idx = None
10     end_idx = None
11
12     for i in range(len(data)):
13         if data[i] == "[":
14             start_idx = i
15             break
16
17     for i in range(len(data)-1, -1, -1):
18         if data[i] == "]":
19             end_idx = i + 1
20             break
21
22     return data[start_idx:end_idx]
23
24
25  def parse_lessons_from_week(data):
26     lessons = []
27
28     start_idx = None
29     end_idx = None
30
31     for i in range(len(data)):
32         if data[i] == "{":
33             start_idx = i
34
35         elif data[i] == "}":
36             end_idx = i + 1
37             lessons.append(data[start_idx:end_idx])
38
39     return lessons
40
```



```

42 def parse_data_from_lesson(data):
43     d = {}
44
45     splitted = data[1:-1].replace(", ", "").split('')
46
47     cleaned_list = []
48     for el in splitted:
49         if el != '' and el != ': ':
50             cleaned_list.append(el)
51
52     for i in range(0, len(cleaned_list), 2):
53         d[cleaned_list[i]] = cleaned_list[i+1]
54
55     return d
56
57
58 def parse_json(data):
59     data = "".join([el.strip() for el in data.splitlines()])
60
61     data = data.split("week")
62
63     even_week = delete_trash(data[1])
64     odd_week = delete_trash(data[2])
65
66     even_week_lessons = parse_lessons_from_week(even_week)
67     even_week_lessons_parsed = [parse_data_from_lesson(lesson) for lesson in even_week_lessons]
68
69     odd_week_lessons = parse_lessons_from_week(odd_week)
70     odd_week_lessons_parsed = [parse_data_from_lesson(lesson) for lesson in odd_week_lessons]
71
72     schedule = {
73         "even week": even_week_lessons_parsed,
74         "odd week": odd_week_lessons_parsed
75     }
76
77     return schedule
78

```

```

80 def json_to_xml(json):
81     xml = "<schedule>\n"
82
83     for i in range(len(json["even week"])):
84         xml += f"<lesson_{i+1}>\n"
85
86         for j in range(len(list(json["even week"][i].keys()))):
87             xml += f"<{list(json['even week'][i].keys())[j].replace(' ', '')}>{list(json['even week'][i].values())[j]}</{list(json['even week'][i].keys())[j].replace(' ', '')}>\n"
88
89         xml += f"</lesson_{i+1}>\n"
90
91     xml += "</even_week>\n"
92
93     for i in range(len(json["odd week"])):
94         xml += f"<lesson_{i+1}>\n"
95
96         for j in range(len(list(json["odd week"][i].keys()))):
97             xml += f"<{list(json['odd week'][i].keys())[j].replace(' ', '')}>{list(json['odd week'][i].values())[j]}</{list(json['odd week'][i].keys())[j].replace(' ', '')}>\n"
98
99         xml += f"</lesson_{i+1}>\n"
100
101     xml += "</odd_week>\n"
102
103     xml += "</schedule>\n"
104
105     return xml
106
107
108 def convert(f1: TextIO, f2: TextIO) -> NoReturn:
109     data = f1.read()
110
111     json = parse_json(data)
112
113     xml = json_to_xml(json)
114
115     f2.write(xml)
116

```

```

117
118 if __name__ == "__main__":
119     f1 = open(F1_NAME, "r", encoding="utf8")
120     f2 = open(F2_NAME, "w", encoding="utf8")
121
122     try:
123         convert(f1, f2)
124
125     except Exception as e:
126         print(f"An error occurred: {e}")
127
128     else:
129         print("Done!")
130
131     finally:
132         f1.close()
133         f2.close()
134

```

C:\Users\n2811\Desktop\itmo\1st_semester\informatics\lab2>python json2xml_by_hands.py
Done!

```

1  <schedule>
2      <even_week>
3          <lesson_1>
4              <name>LIFE SAFETY (LEK): INTRODUCTORY LECTURE</name>
5              <teacher>Rusanov Dmitriy Yurevich</teacher>
6              <type>distant</type>
7              <day>monday</day>
8              <start_time>18:40</start_time>
9              <end_time>20:10</end_time>
10         </lesson_1>
11     </even_week>
12     <lesson_1>
13         <name>LIFE SAFETY (LEK): INTRODUCTORY LECTURE</name>
14         <teacher>Rusanov Dmitriy Yurevich</teacher>
15         <type>distant</type>
16         <day>monday</day>
17         <start_time>18:40</start_time>
18         <end_time>20:10</end_time>
19     </lesson_1>
20     <lesson_2>
21         <name>LIFE SAFETY (LEK): INTRODUCTORY LECTURE</name>
22         <teacher>Rusanov Dmitriy Yurevich</teacher>
23         <type>distant</type>
24         <day>monday</day>
25         <start_time>18:40</start_time>
26         <end_time>20:10</end_time>
27     </lesson_2>
28 </odd_week>
29 </schedule>

```

Дополнительное задание №1

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

Решение

```
1  import json
2  from time import time
3
4  from dicttoxml import dicttoxml
5
6  from json2xml_by_hands import *
7
8
9  F1_NAME = "schedule.json"
10 F2_NAME = "schedule.xml"
11
12
13 def convert_by_lib(f1, f2):
14     data = f1.read()
15
16     xml = dicttoxml(json.loads(data))
17
18     f2.write(str(xml))
19
```

```

22 if __name__ == "__main__":
23     f1 = open(F1_NAME, "r", encoding="utf8")
24     f2 = open(F2_NAME, "w", encoding="utf8")
25
26     try:
27         start_time_hands = time()
28         convert(f1, f2)
29         end_time_hands = time()
30
31         f1.close()
32         f2.close()
33
34         f1 = open(F1_NAME, "r", encoding="utf8")
35         f2 = open(F2_NAME, "w", encoding="utf8")
36
37         start_time_lib = time()
38         convert_by_lib(f1, f2)
39         end_time_lib = time()
40
41         hands_time = end_time_hands - start_time_hands
42         lib_time = end_time_lib - start_time_lib
43
44         print(f"My parser time: {hands_time}")
45         print(f"Lib parser time: {lib_time}")
46
47     except Exception as e:
48         print(f"An error occurred: {e}")
49
50     else:
51         print("Done!")
52
53     finally:
54         f1.close()
55         f2.close()
56

```

```

C:\Users\n2811\Desktop\itmo\1st_semester\informatics\lab2>python json2xml_by_libs.py
My parser time: 0.0009980201721191406
Lib parser time: 0.00498199462890625
Done!

```

Мой перевод работает быстрее, но он работает только с данной архитектурой расписания и не является полностью универсальным решением

Вывод

Я освоил python, регулярные выражения, форму Бэкуса-Наура, особенности json, yaml, xml, а также научился переводить из одного формата в другой самостоятельно.

Список литературы

1. Балакшин П.В., Соснин В.В., Машина Е.А. Информатика