НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Факультет ПИиКТ

Дисциплина

Лабораторная работа № 3

Выполнил студент

Набокова Алиса Владиславовна

Группа № 3120

Преподаватель: Болдырева Елена Александровна

г. Санкт-Петербург

2023

Оглавление

Отчёт:..... 3
Задание 1..... 3
Задание 2..... 4
Задание 3..... 5
Вывод: 7
Список литературы: 7

Отчёт:

Задание 1

Вариант: 135

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 в предложенном тексте.

Смайлик: ;<{/

Код:

```
import re

def count_smileys(text):
    pattern = r";<{/"  # Регулярное выражение для поиска смайлика
    matches = re.findall(pattern, text)  # Находим все совпадения с регулярным выражением в тексте
    return len(matches)  # Возвращаем количество найденных смайликов

# Тесты

# Тест 1: текст содержит 2 смайлика
text1 = "Привет! ;<{/ Это мой новый смайлик ;<{/ Как дела?"
expected1 = 2
result1 = count_smileys(text1)
print(f"Тест 1: Ожидаемый результат: {expected1}, Полученный результат: {result1}")
print()

# Тест 2: текст содержит несколько смайликов подряд
text2 = "Смайлики смайлики ;<{/ ;<{/ ;<{/ "
expected2 = 3
result2 = count_smileys(text2)
print(f"Тест 2: Ожидаемый результат: {expected2}, Полученный результат: {result2}")
print()

# Тест 3
text3 = "Хорошего дня, не обращай внимание на недовольный смайлик, он просто не успел сегодня позавтракать ;<{/ "
expected3 = 1
result3 = count_smileys(text3)
print(f"Тест 3: Ожидаемый результат: {expected3}, Полученный результат: {result3}")
print()

# Тест 4: текст не содержит смайликов
text4 = "Это просто обычный текст без смайликов"
expected4 = 0
```

```python
# Тест 5
text5 = "Текст про удивлённого усача, которого никто не замечает ;<{0/ ;<{/ "
expected5 = 1
result5 = count_smileys(text5)
print(f"Тест 5: Ожидаемый результат: {expected5}, Полученный результат: {result5}")
```

Вывод программы:

```
Тест 1: Ожидаемый результат: 2, Полученный результат: 2

Тест 2: Ожидаемый результат: 3, Полученный результат: 3

Тест 3: Ожидаемый результат: 1, Полученный результат: 1

Тест 4: Ожидаемый результат: 0, Полученный результат: 0

Тест 5: Ожидаемый результат: 1, Полученный результат: 1

Process finished with exit code 0
```

Задание 2

Вариант: 5

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.

Анатолий выложил пост с расписанием доп. занятий по информатике, но везде перепутал время. Поэтому нужно заменить все вхождения времени на строку (TBD).
Время – это строка вида HH:MM:SS или HH:MM, в которой HH – число от 00 до 23, а MM и SS – число от 00 до 59.

Код:

```python
import re

def replace_time(post):
    pattern = r"([01][0-9]|2[0-3])(:[0-5][0-9])(:([0-5][0-9]))?"  # Регулярное выражение для поиска времени
    replaced_post = re.sub(pattern, "(TBD)", post)  # Замена времени на "(TBD)"
    return replaced_post

# Тесты

# Тест 1
post1 = "Доп. занятие по информатике будет в 09:30:00, а не в 10:00:00"
expected1 = "Доп. занятие по информатике будет в (TBD), а не в (TBD)"
result1 = replace_time(post1)
print(f"Тест 1: Ожидаемый результат: {expected1}, Полученный результат: {result1}")
print()

# Тест 2
post2 = "Занятие начнется в 15:30, а не в 16:00"
expected2 = "Занятие начнется в (TBD), а не в (TBD)"
result2 = replace_time(post2)
print(f"Тест 2: Ожидаемый результат: {expected2}, Полученный результат: {result2}")
print()

# Тест 3
post3 = "Уважаемые студенты! В эту субботу в 15:00 планируется доп. занятие на 2 часа. То есть в 17:00:01 оно уже точно кончится."
expected3 = "Уважаемые студенты! В эту субботу в (TBD) планируется доп. занятие на 2 часа. То есть в (TBD) оно уже точно кончится."
result3 = replace_time(post3)
print(f"Тест 3: Ожидаемый результат: {expected3}, Полученный результат: {result3}")
print()
```

```python
# Тест 4
post4 = "Встреча в 10:00 будет перенесена на 11:30"
expected4 = "Встреча в (TBD) будет перенесена на (TBD)"
result4 = replace_time(post4)
print(f"Тест 4: Ожидаемый результат: {expected4}, Полученный результат: {result4}")
print()

# Тест 5
post5 = "Семинар будет проводиться в 13:30 и закончится в 15:30"
expected5 = "Семинар будет проводиться в (TBD) и закончится в (TBD)"
result5 = replace_time(post5)
print(f"Тест 5: Ожидаемый результат: {expected5}, Полученный результат: {result5}")
```

Вывод программы:

```
Тест 1: Ожидаемый результат: Доп. занятие по информатике будет в (TBD), а не в (TBD), Полученный результат: Доп. занятие по информатике будет в (TBD), а не в (TBD)

Тест 2: Ожидаемый результат: Занятие начнется в (TBD), а не в (TBD), Полученный результат: Занятие начнется в (TBD), а не в (TBD)

Тест 3: Ожидаемый результат: Уважаемые студенты! В эту субботу в (TBD) планируется доп. занятие на 2 часа. То есть в (TBD) оно уже точно кончится., Полученный резуль

Тест 4: Ожидаемый результат: Встреча в (TBD) будет перенесена на (TBD), Полученный результат: Встреча в (TBD) будет перенесена на (TBD)

Тест 5: Ожидаемый результат: Семинар будет проводиться в (TBD) и закончится в (TBD), Полученный результат: Семинар будет проводиться в (TBD) и закончится в (TBD)

Process finished with exit code 0
```

Задание 3

Вариант: 35

1) Исходя из структуры расписания конкретного дня, сформировать файл с расписанием в формате, указанном в задании в качестве исходного.

2)Написать программу на языке Python 3.x, которая бы осуществляла парсинг и конвертацию исходного файла в новый.

3)Нельзя использовать готовые библиотеки, в том числе регулярные выражения в Python и библиотеки для загрузки XML-файлов.

XML файл:

```xml
<saturday>
    <lessons>
        <firstLesson>
            <time>17:00-18:30</time>
            <place>Кронверский пр., д.49, лит.А 1410 ауд.</place>
            <nameLesson>ПРАКТИКУМ ПО BACKEND-РАЗРАБОТКЕ НА ЯЗЫКЕ GO</nameLesson>
            <typeActivity>Практика</typeActivity>
            <typeMeeting>Очно-дистанционный</typeMeeting>
        </firstLesson>
        <secondLesson>
            <time>18:40-20:10</time>
            <place>Кронверский пр., д.49, лит.А 1410 ауд.</place>
            <nameLesson>АЛГОРИТМЫ И СТРУКТУРЫ ДАННЫХ</nameLesson>
            <typeActivity>Лекция</typeActivity>
            <typeMeeting>Очно-дистанционный</typeMeeting>
        </secondLesson>
        <thirdLesson>
            <time>20:20-21:50</time>
            <place>Кронверский пр., д.49, лит.А 1410 ауд.</place>
            <nameLesson>ЯЗЫКИ ПРОГРАММИРОВАНИЯ</nameLesson>
            <typeActivity>Практика</typeActivity>
            <typeMeeting>Очно-дистанционный</typeMeeting>
        </thirdLesson>
    </lessons>
</saturday>
```

Код:

```python
yaml = []
file = open('xml.input.xml')
strings = file.readlines()
for i in range(len(strings)):

    strings[i] = strings[i].replace('<', '')
    strings[i] = strings[i].replace('>', ': ')

    if '/' in strings[i]:
        strings[i] = strings[i][:strings[i].find('/')]
    if ': ' in strings[i]:
        strings[i] = strings[i][:strings[i].find(':')+2]+'"'+strings[i][strings[i].find(':')+2:]+'"'
    if '\n' in strings[i]:
        strings[i] = strings[i][:strings[i].find(': ')+1]

    yaml.append(strings[i])

print(*yaml, sep="\n")
```

Вывод программы:

```
saturday:
    lessons:
        firstLesson:
            time: "17:00-18:30"
            place: "Кронверский пр., д.49, лит.А 1410 ауд."
            nameLesson: "ПРАКТИКУМ ПО BACKEND-РАЗРАБОТКЕ НА ЯЗЫКЕ GO"
            typeActivity: "Практика"
            typeMeeting: "Очно-дистанционный"

        secondLesson:
            time: "18:40-20:10"
            place: "Кронверский пр., д.49, лит.А 1410 ауд."
            nameLesson: "АЛГОРИТМЫ И СТРУКТУРЫ ДАННЫХ"
            typeActivity: "Лекция"
            typeMeeting: "Очно-дистанционный"

        thirdLesson:
            time: "20:20-21:50"
            place: "Кронверский пр., д.49, лит.А 1410 ауд."
            nameLesson: "ЯЗЫКИ ПРОГРАММИРОВАНИЯ"
            typeActivity: "Практика"
            typeMeeting: "Очно-дистанционный"
```

Вывод:
В ходе проделанной работы были изучены регулярные выражения, языки разметки: xml, json, yaml, а также форма Бэкуса-Наура

Список литературы:

Лямин А.В., Череповская Е.Н. Объектно-ориентированное программирование. Компьютерный практикум. – СПб: Университет ИТМО, 2017. – 143 с. – Режим доступа: https://books.ifmo.ru/file/pdf/2256.pdf.

URL: https://ru.wikipedia.org/wiki/Форма_Бэкуса_—_Наура.

URL: https://habr.com/ru/post/309242/.

URL: https://habr.com/ru/articles/349860/