Студент группы ИС-25 Несповитый А.Е.

Практическое занятие №14

Тема: составление программ с использованием регулярных выражений в IDE PyCharm Community.

Цель: закрепить усвоенные знания, понятия, алгоритмы,
основные принципы составления программ, приобрести навыки составление программ с использованием с использованием регулярных выражений в IDE PyCharm Community.

Постановка дополнительной задачи:
https://regex101.com/r/aGn8QC/2

- Найдите все натуральные числа (возможно, окружённые буквами);
- Найдите все «слова», написанные капсом (то есть строго заглавными), возможно внутри настоящих слов (аааБББввв);
- Найдите слова, в которых есть русская буква, а за ней цифра;
- Найдите все слова, начинающиеся с русской или латинской большой буквы (\b — граница слова);
- Найдите слова, которые начинаются на гласную (\b — граница слова);
- Найдите все натуральные числа, не находящиеся на границе слова;
- Найдите строчки, в которых есть символ * (. — это точно не конец строки!);
- Найдите строчки, в которых есть открывающая и когда-нибудь потом закрывающая скобки;
- Выделите одним махом весь кусок оглавления (в конце примера, вместе с тегами);
- Выделите одним махом только текстовую часть оглавления, без тегов;
- Найдите пустые строчки;
- Найти все теги, не включая их содержимое.

Тип алгоритма: Циклический

Текст программы:

```
"""
https://regex101.com/r/aGn8QC/2
•	Найдите все натуральные числа (возможно, окружённые буквами);
•	Найдите все «слова», написанные капсом (то есть строго заглавными), возможно внутри настоящих слов (аааБББввв);
•	Найдите слова, в которых есть русская буква, а за ней цифра;
•	Найдите все слова, начинающиеся с русской или латинской большой буквы (\b — граница слова);
•	Найдите слова, которые начинаются на гласную (\b — граница слова);
•	Найдите все натуральные числа, не находящиеся на границе слова;
•	Найдите строчки, в которых есть символ * (. — это точно не конец строки!);
•	Найдите строчки, в которых есть открывающая и когда-нибудь потом закрывающая скобки;
•	Выделите одним махом весь кусок оглавления (в конце примера, вместе с тегами);
•	Выделите одним махом только текстовую часть оглавления, без тегов;
•	Найдите пустые строчки;
•	Найти все теги, не включая их содержимое.
"""


import re

patterns = [
    r"\d+",
    r"[А-ЯЁ]+",
    r"\b[а-яё]\d+\b",
    r"\b[A-ЯЁA-Z]\w+\b",
    r"\b[аеёиоуыэюяАЕЁИОУЫЭЮЯ]\w+\b",
    r"(?<!\b)\d+(?!\b)",
    r".*\*.*",
```

```
    r".*\(.+\).*",
    r"<ul>(.*)</ul>",
    r"<ul>(.*?)</ul>",
    r"^$",
    r"<.*?>",
]

pattern_names = [
    "Найти все натуральные числа",
    "Найти все слова, написанные капсом",
    "Найти слова, в которых есть русская буква, а за ней цифра",
    "Найти все слова, начинающиеся с русской или латинской большой буквы",
    "Найти слова, которые начинаются на гласную",
    "Найти все натуральные числа, не находящиеся на границе слова",
    "Найти строчки, в которых есть символ *",
    "Найти строчки, в которых есть открывающая и когда-нибудь потом закрывающая
скобки",
    "Выделите одним махом весь кусок оглавления",
    "Выделите одним махом только текстовую часть оглавления",
    "Найти пустые строчки",
    "Найти все теги, не включая их содержимое",
]


def find_pattern(text, pattern_number):
    """
    Функция для поиска текста по регулярному выражению,
    принимает номер шаблона и текст.

    Args:
      text (str): Текст для поиска.
      pattern_number (int): Номер шаблона.

    Returns:
      list: Список найденных совпадений.
    """

    if pattern_number > len(patterns):
        raise ValueError("Номер шаблона должен быть от 1 до {}".format(len(patterns)))

    pattern = patterns[pattern_number - 1]
    return re.findall(pattern, text)


def main():
    """
    Функция для отображения меню и выполнения поиска.
    """

    while True:
        for i, name in enumerate(pattern_names, start=1):
            print("{}. {}".format(i, name))
        print("Выберите номер шаблона (1-{}) или 0 для выхода: ".format(len(patterns)))

        try:
            choice = int(input())
        except ValueError:
            print("Неверный формат. Введите число.")
            continue

        if choice == 0:
            break
```

```
        if choice < 1 or choice > len(patterns):
            print("Номер шаблона должен быть от 1 до {}.".format(len(patterns)))
            continue

        # Пример использования
        text = """
        Привет, мир!

        123abc456
        АБВГДЕёЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ
        апав
        агав

        Привет123

        Привет, Мир!

        *важное сообщение*

        (скобки)

        <ul>
          <li>Пункт 1</li>
          <li>Пункт 2</li>
        </ul>

        """
        print("**{}:**".format(pattern_names[choice - 1]))
        print(find_pattern(text, choice))

        print('Хотите продолжить нажмите 1, если хотите закончить то 0: ')
        if int(input()) == 1:
            continue
        else:
            break


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

Протокол программы:
1. Найти все натуральные числа
2. Найти все слова, написанные капсом
3. Найти слова, в которых есть русская буква, а за ней цифра
4. Найти все слова, начинающиеся с русской или латинской большой буквы
5. Найти слова, которые начинаются на гласную
6. Найти все натуральные числа, не находящиеся на границе слова
7. Найти строчки, в которых есть символ *
8. Найти строчки, в которых есть открывающая и когда-нибудь потом закрывающая скобки
9. Выделите одним махом весь кусок оглавления
10. Выделите одним махом только текстовую часть оглавления
11. Найти пустые строчки
12. Найти все теги, не включая их содержимое
Выберите номер шаблона (1-12) или 0 для выхода:
5
Найти слова, которые начинаются на гласную:
['АБВГДЕёЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ', 'апав', 'агав']

Хотите продолжить нажмите 1, если хотите закончить то 0:
1
1. Найти все натуральные числа
2. Найти все слова, написанные капсом
3. Найти слова, в которых есть русская буква, а за ней цифра
4. Найти все слова, начинающиеся с русской или латинской большой буквы
5. Найти слова, которые начинаются на гласную
6. Найти все натуральные числа, не находящиеся на границе слова
7. Найти строчки, в которых есть символ *
8. Найти строчки, в которых есть открывающая и когда-нибудь потом закрывающая скобки
9. Выделите одним махом весь кусок оглавления
10. Выделите одним махом только текстовую часть оглавления
11. Найти пустые строчки
12. Найти все теги, не включая их содержимое
Выберите номер шаблона (1-12) или 0 для выхода:
6
Найти все натуральные числа, не находящиеся на границе слова:
['23', '45', '12']
Хотите продолжить нажмите 1, если хотите закончить то 0:
1
1. Найти все натуральные числа
2. Найти все слова, написанные капсом
3. Найти слова, в которых есть русская буква, а за ней цифра
4. Найти все слова, начинающиеся с русской или латинской большой буквы
5. Найти слова, которые начинаются на гласную
6. Найти все натуральные числа, не находящиеся на границе слова
7. Найти строчки, в которых есть символ *
8. Найти строчки, в которых есть открывающая и когда-нибудь потом закрывающая скобки
9. Выделите одним махом весь кусок оглавления
10. Выделите одним махом только текстовую часть оглавления
11. Найти пустые строчки
12. Найти все теги, не включая их содержимое
Выберите номер шаблона (1-12) или 0 для выхода:
1
Найти все натуральные числа:
['123', '456', '123', '1', '2']
Хотите продолжить нажмите 1, если хотите закончить то 0:
0

Process finished with exit code 0

Постановка задачи №1:

```
Из исходного текстового файла (ip_address.txt) из раздела «Частоупотребимые
маски» перенести в первый файл строки с нулевым четвертым октетом, а во второй
– все остальные. Посчитать количество полученных строк в каждом файле.
```

Тип алгоритма: Циклический
Текст программы:

```
"""
Из исходного текстового файла (ip_address.txt) из раздела «Частоупотребимые
маски» перенести в первый файл строки с нулевым четвертым октетом, а во второй
– все остальные. Посчитать количество полученных строк в каждом файле.
"""
import re
```

```python
def find_text(text):
    """
    Args:
      text: Текст, в котором нужно найти текст.
      start_marker: Начало поиска
      end_marker: Конец поиска

    Returns:
      Строка с найденным текстом.
    """
    start_marker = "Частоупотребимые маски"
    end_marker = "Количество адресов подсети не равно количеству возможных узлов. Нулевой IP-адрес резервируется для идентификации подсети, последний — в качестве широковещательного адреса. Таким образом, в реально действующих сетях возможно количество узлов на два меньшее количества адресов."

    start_index = text.find(start_marker)
    if start_index == -1:
        return None

    end_index = text.find(end_marker, start_index + len(start_marker))
    if end_index == -1:
        return None

    return text[start_index + len(start_marker):end_index]


with open("records/ip_address.txt", "r") as input_file, \
        open("records/first_file.txt", "w") as first_output_file, \
        open("records/second_file.txt", "w") as second_output_file:
    for line in find_text(input_file.read()).split('\n'):
        # Проверка четвертого октета
        if re.match(r".*\.0$", line):
            first_output_file.write(line + '\n')
        elif not line in ' ':
            second_output_file.write(line + '\n')

# Подсчет строк
first_file_lines = sum(1 for line in open("records/first_file.txt"))
second_file_lines = sum(1 for line in open("records/second_file.txt"))

print("Внутреннее содержание файла first_file.txt:",
open("records/first_file.txt").read().split())
print(f"Количество строк в first_file.txt: {first_file_lines}\n")
print("Внутреннее содержание файла second_file.txt:",
open("records/second_file.txt").read().split())
print(f"Количество строк в second_file.txt: {second_file_lines}\n")
```

Протокол программы:
Внутреннее содержание файла first_file.txt: ['255.255.128.0', '255.255.192.0', '255.255.224.0', '255.255.240.0', '255.255.248.0', '255.255.252.0', '255.255.254.0', '255.255.255.0']
Количество строк в first_file.txt: 8

Внутреннее содержание файла second_file.txt: ['255.255.255.128', '255.255.255.192', '255.255.255.224', '255.255.255.240', '255.255.255.248', '255.255.255.252']
Количество строк в second_file.txt: 6

Process finished with exit code 0

Вывод: в процессе выполнения практического занятия закрепил усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрести навыки составление программ с использованием с использованием регулярных выражений в IDE PyCharm