Федеральное государственное автономное образовательное
учреждение высшего образования
«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Отчёт по лабораторной работе №3

По дисципление: Информатика

Тема: Регулярные выражения

Вариант (по ИСУ) 467213

Выполнил: Разыграев Кирилл Сергеевич

Группа: P3115

Преподаватель: Белокон Юлия Алексеевна

Санкт-Петербург, 2024

Задание 1

Задание

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. Все 5 тестов необходимо показать при защите.

3. Программа должна считать число смайликов определённого вида (вид смайлика описан в таблице вариантов) в предложенном тексте. Все смайлики имеют такую структуру: [глаза][нос][рот]. Вариантом является различные наборы глаз, носов и ртов.

Номер в ИСУ % 6	Глаза	Номер в ИСУ % 4	Нос	Номер в ИСУ % 8	Рот
0	8	0	-	0	(
1	;	1	<	1	)
2	X	2	-{	2	P
3	:	3	<{	3	|
4	=			4	\
5	[			5	/
				6	O
				7	=

Тестовые файлы

Тест 1

Lorem [</ ipsum odor amet, consectetuer [</adipiscing elit.

Тест 2

Cra[</s eu auctor mus, ve[</hicula mass[</a ligula.

Тест 3

Nulla pellentesque turpis [</ dictum accumsan tortor ac.

Тест 4

Sit senectus acc[</umsan ultricies ac nasce[</tur mus quis taciti

Тест 5

In curabitur ante dolor; torquent[</ erat [</sagittis [</cras fermentu[</m.

Вычислятор смайликов

```
EYES = ("8", ";", "X", ":", "=", "[")
NOSE = ("-", "<", "-{", "<{")
MOUTH = ("(", ")", "P", "|", "\\", "/", "O", "=")
^^I^^I
^^I^^I
def get_isu() -> int:
    raw = input("Введите свой ISU: ")
    if not raw.isdigit():
        raise ValueError("ISU должен состьять из цифр")
    if len(raw) != 6:
        raise ValueError("ISU должен состоять из 6 цифр")

    return int(raw)

^^I^^I
def calculate_emoji(isu: int) -> str:
    emoji = EYES[isu % 6] + NOSE[isu % 4] + MOUTH[isu % 8]

    return emoji
^^I^^I
```

```
^^I^^I
def main() -> None:
    isu = get_isu()
    emoji = calculate_emoji(isu)
^^I^^I
    print(f"Ваш смайлик: {emoji}")
^^I^^I
^^I^^I
if __name__ == "__main__":
    main()
```

Листинг

```
import re
from pathlib import Path

TEXTS_PATH = Path(__file__).parent / "texts" / "task_1"
EMOJI_PATTERN = re.compile(r"\[</")

CORRECT_COUNTS = (1, 3, 1, 2, 4)


def count_emoji(text: str) -> int:
    return len(EMOJI_PATTERN.findall(text))


def main() -> None:
    for i in range(1, 6):
        text = (TEXTS_PATH / f"text_{i}").read_text()
        emoji_count = count_emoji(text)

        if emoji_count != CORRECT_COUNTS[i - 1]:
            print(
                f"Ошибка в тесте {i}. "
                f"Ожидалось - {CORRECT_COUNTS[i - 1]}, "
                f"найдено - {emoji_count}"
            )
        else:
            print(f"Текст {i}: смайликов - {emoji_count}")
```

```
        print("=" * 100)


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

Задание 2

Задание

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. Все 5 тестов необходимо показать при защите. Пример тестов приведён в таблице.
3. Можно использовать циклы и условия, но основной частью решения должны быть регулярные выражения.

5	С помощью регулярного выражения найти в тексте все слова, в которых две гласные стоят подряд, а после этого слова идёт слово, в котором не больше 3 согласных. Пример: Ввод: Кривошеее существо гуляет по парку Вывод: гуляет

Тестовые файлы

Тест 1

Кривошеее существо гуляет по парку

Тест 2

Кривоногое создание с интересом рассматривало яркое солнце над горизонтом

Тест 3

Синее небо раскрылось перед маленьким озером, окруженным камнями. Двуххвостое животное лениво отдыхало под тихим ветром на берегу моря.

Тест 4

Безухое создание медленно пробиралось через тёплую траву на поляне.

Тест 5

Пятиногое создание медленно двигалось через зелёную поляну у леса.

Листинг

```
import re
from pathlib import Path

TEXTS_PATH = Path(__file__).parent / "texts" / "task_2"
PATTERN = re.compile(
    r"\b\w*[аеёиоуыэюя]{2}\w*\b(?!\s+(?:[аеёиоуыэюя]*[бвгджзйклмнпрстфхцчшщ]){4,})",
    flags=re.MULTILINE
)
CORRECT_ANSWERS = (
    {"гуляет"},
    {"создание"},
    {"Синее", "животное"},
    {"тёплую"},
    {"зелёную"}
)
```

```python
def find_words(text: str) -> set[str]:
    return set(PATTERN.findall(text))


def main() -> None:
    for i in range(1, 6):
    text = (TEXTS_PATH / f"text_{i}").read_text(encoding="utf-8")
    words = find_words(text)

    if words != CORRECT_ANSWERS[i - 1]:
        print(
            f"Ошибка в тесте {i}. "
            f"Ожидалось - {CORRECT_ANSWERS[i - 1]}, "
            f"Результат - {words}"
        )
    else:
        print(f"Текст {i}: слова - {words}")
    print("=" * 100)


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

Задание 3

Задание

1. Для своей программы придумайте минимум 5 тестов. Все 5 тестов необходимо показать при защите.
2. Протестируйте свою программу на этих тестах
3. Можно использовать циклы и условия, но основной частью решения должны быть регулярные выражения.

5	Вывесили списки стипендиатов текущего семестра, которые представляют из себя список людей ФИО и номер группы этого человека. Вы решили подшутить над некоторыми из своих одногруппников и удалить их из списка. С помощью регулярного выражения найдите всех студентов своей группы, у которых инициалы начинаются на одну и туже букву и исключите их из списка. Могут существовать двойные фамилии, которые тоже нужно учитывать (студенты с такими фамилиями тоже должны иметь право быть удаленными из списка стипендиатов текущего семестра). Пример (группа P0000): Ввод: Петров П.П. P0000 Анищенко А.А. P33113 Примеров Е.В. P0000 Иванов И.И. P0000 Вывод: Анищенко А.А. P33113 Примеров Е.В. P0000

Тестовые файлы

Тест 1

Петров П.П. P0000

Петров П.П. P3115

Анищенко A.Z. P3115

Тест 2

Сидоров С.С. P3115

Петров П.П. P3114

Васильев В.И. P3115

Тест 3

Кузнецов А.А. P3115

Лебедев К.Н. P3116

Тест 4

Сафонов С.Ю. P3115

Соловьев В.А. P3115

Тест 5

Романов Р.С. Р3115

Попов П.П. Р3115

Листинг

```
import re
from pathlib import Path

GROUP = "P3115"

TEXTS_PATH = Path(__file__).parent / "texts" / "task_3"
PATTERN = re.compile(
    r"^(\w)\w+\s+(?:\1(?:\.|\w+\s+))+\s+{group}$".format(group=GROUP),
    flags=re.IGNORECASE | re.MULTILINE
)

CORRECT_ANSWERS = (
    "Петров П.П. Р0000\nАнищенко А.Z. Р3115",
    "Петров П.П. Р3114\nВасильев В.И. Р3115",
    "Кузнецов А.А. Р3115\nЛебедев К.Н. Р3116",
    "Сафонов С.Ю. Р3115\nСоловьев В.А. Р3115",
    "Романов Р.С. Р3115"
)


def find_words(text: str) -> set[str]:
    return set(PATTERN.findall(text))


def main() -> None:
    for i in range(1, 6):
        text = (TEXTS_PATH / f"text_{i}").read_text(encoding="utf-8")
        processed_text = PATTERN.sub("", text).strip()
        processed_text = re.sub(r"\n{2,}", "\n", processed_text)
        correct_answer = CORRECT_ANSWERS[i - 1]

        if processed_text != correct_answer:
            print(
```

```
                f"Ошибка в тесте {i}\n"
                f"Ожидалось - {correct_answer}\n"
                f"Результат - {processed_text}"
            )
        else:
            print(f"Текст {i}: результат - {processed_text}")
            print("=" * 100)


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