работа на уроке (28.04)

1. Напишите функцию `texts_similarity(left, rigth)`, принимающую 2 текста и возвращающую числовую оценку похожести этих 2 текстов.
   Для оценки похожести используйте следующую формулу:

   ```
   0.4 * position_mark + 0.6 * distinct_mark
   ```

   где:

   ```
   position_mark =
       <количество совпадающих слов, стоящих на одних и тех же позициях в 2 текстах> /
       <среднее количество слов в 2 текстах>
   ```

   ```
   distinct_mark =
       <количество уникальных слов, которые есть в обоих текстах> /
       <общее количество уникальных слов в 2 текстах>
   ```

   NB! Ваше решение **не должно состоять из одной функции**. Разбейте задачу на небольшие куски и напишите для каждого куска свою функцию. Затем объедините все в единую программу.

Пример

```
left_text = 'Иван рубил дрова'
right_text = 'Семен рубил дрова и капусту'
mark = texts_similarity(left_text, right_text)
print(mark)
0.38
```

Объяснение

```
<среднее количество слов в 2 текстах> = 4
<количество совпадающих слов, стоящих на одних и тех же позициях в 2 текстах> = 2
position_mark = 0.5

<количество уникальных слов, которые есть в обоих текстах> = 2 (рубил, дрова)
<общее количество уникальных слов в 2 текстах> = 6
distinct_mark = 0.3

mark = 0.4 * 0.5 + 0.6 * 0.3 = 0.38
```

2. Вам нужно написать функцию, которая находит в тексте упоминания заданного слова и возвращает контекст, в котором это слово употреблялось. Само слово должно быть возвращено в верхнем регистре.

   В качестве контекста нужно возвращать строку, на которой было найдено слово и по одной строке сверху и снизу от строки с найденным словом. Оформить это можно в виде **списка** из 3 элементов. Или кортежа. Нужное слово может употребляться в тексте несколько раз. Вам нужно найти все упоминания. Ваша функция должна возвращать список списков(кортежей).

   - если строка, сверху или снизу от строки с найденным словом пустая, то нужно взять первую непустую строку.
   - если строка с найденным словом последняя или первая, то соответствующий контекст сверху и/или снизу должны иметь значение `None`.

   NB! Ваше решение **не должно состоять из одной функции**. Разбейте задачу на небольшие куски и напишите для каждого куска свою функцию. Затем объедините все в единую программу. Например, эту задачу можно разбить на 4 логических куска: поиск строк, где упоминается введенное слово; нахождение первой непустой строки сверху от найденной строки; нахождение непустой строки снизу от найденной строки; приведение нужного слова в верхний регистр.

   Пример

```
text = '''
Не ставя под сомнение возможность разных подходов к почве,
бугор пучения отталкивает лизиметр. Копролит сжимает подбел.
При переходе к следующему уровню организации почвенного покрова
ожелезнение локально притягивает поглотительный
псевдомицелий. Почвенная влага, как того требуют законы термодинамики,
стекает в иловатый ортштейн. В соответствии с
принципом неопределенности, реология эволюционирует в внутрипочвенный электрод.
```

```
Кора выветривания снижает иловатый влагомер. Почвообразовательный процесс
представляет собой глей. Эпюра традиционно притягивает лизиметр, что дает
возможность использования данной методики как универсальной. Плотность
твёрдой фазы восстанавливает шурф.

Монолит усиливает разрез. Ожелезнение, по данным почвенной съемки,
пространственно приводит к появлению вязкий влагомер. В первом
приближении пористость доступна.
'''

result = find_word_with_context(text, 'влагомер')
print(result)
[
    [
        'принципом неопределенности, реология эволюционирует в внутрипочвенный электр
од.',
        'Кора выветривания снижает иловатый ВЛАГОМЕР. Почвообразовательный процесс',
        'представляет собой глей. Эпюра традиционно притягивает лизиметр, что дает'
    ],
    [
        'Монолит усиливает разрез. Ожелезнение, по данным почвенной съемки,',
        'пространственно приводит к появлению вязкий ВЛАГОМЕР. В первом',
        'приближении пористость доступна.'
    ]
]
```

3. Напишите функцию, которая проверяет правописание чередующихся корней: ЛАГ/ЛОЖ, РАСТ/РАЩ/РОС, КАС/КОСН, СКАК/СКОЧ.
   Функция принимает на вход слово (написанное со звездочкой вместо буквы в корне, например, `‘предпол*гать’`), возвращает слово с правильной гласной. Исключения учитывать необязательно, но можно, если есть желание.

4. Напишите функцию `roman_to_int`, которая переводит римские запись числа в арабскую.

   ## Пример:

   ```
   Input: s = "III"
   Output: 3
   ```

   ## Пример:

```
Input: s = "IV"
Output: 4
```

Пример:

```
Input: s = "IX"
Output: 9
```

Пример:

```
Input: s = "LVIII"
Output: 58
```

Объяснение: `L = 50`, `V= 5`, `III = 3`.

Пример:

```
Input: s = "MCMXCIV"
Output: 1994
```

Объяснение: `M = 1000`, `CM = 900`, `XC = 90`, `IV = 4`.

5. Выгрузите любой телеграм-чат в формате `machine-readable JSON`.
   Выясните, кто из участников насколько активен: составьте частотный словарь вида `id или имя участника: кол-во его сообщений`. Сохраните частотный словарь в `json` (обратите внимание на параметры `ensure_ascii` и `indent`)

6. Напишите программу, которая загадывает персонажей “Звёздных войн”.
   Загадав персонажа, программа показывает подсказку в виде одной из трёх самых частотных биграмм из реплик этого персонажа, и ждёт ответа пользователя, после чего сообщает, угадал он или нет. Например, если загадан персонаж «THREEPIO», можно показать подсказку «Master Luke».
   Реплики персонажей нужно брать из сценариев Звёздных войн: ссылка на ZIP-архив сценариев.

7. В условиях задачи 6, реализуйте возможность попробовать угадать персонажа несколько (например, 5) раз и печатайте количество правильных ответов в процентном соотношении.

8. В условиях задачи 6, реализуйте возможность подсказать пользователю факт о персонаже, если пользователь не угадал персонажа: если пользователь не угадал ответ, нужно выдать в ответ какую-то (если есть) информацию о загаданном персонаже из датасета (см. файл `characters.csv`).
   Примечание: во избежание рутины, можно ограничить программу знанием только про пять персонажей на ваш выбор. Соответствие имён персонажей между датасетами задайте вручную (Например, `{"THREEPIO": "C-3PO"}`).