работа на уроке (23.01)

1. `wget.download()` `with open(…) as …:` `.read() / .readlines()` `.split('\t')` `.isalpha()` `string.ascii_letters` `random.choice()` `dict()` `with open(…) as …:` `'w'` `json.dump() / json.dumps()`

   Ник в мессендже Lenagram — это собака, затем от 7 до 9 латинских букв. Допускаются подчеркивания и точки, но не более одного подряд. Если вы хотите дать кому-то свой контакт, то буквы могут быть в любом регистре, т.е. `@LOGIN` и `@login` — это один и тот же человек.

   В CSV-файле (заметьте, это файл с табуляцией `'\t'` в качестве разделителя, то есть TSV-файл) затесались несколько невалидных ников. Найдите валидные ники и выведите из них три случайных (вместе с соответствующими emoji) в JSON-файл подобной структуры:

   ```
   [
   {
    "nick": "@1231__231:,
    "emojis": "💋 💌"
   },
   {
    "nick": "@1__2!@###$1:,
    "emojis": "💌"
   }
   ]
   ```

2. Существует словарь эвенского языка в формате `json`. Цель — создать из этого словарь в формате `dsl`, с которым может работать программа GoldenDict.

   Формат `dsl` выглядит следующим образом: это текстовый файл со словарными статьями и метаразметкой. Слово в начале строки — словарное вхождение (лемма), определение же начинается с табуляции.

   ```
   А̄-А
     [b]А̄-А[/b] междом., крик при легком испуге, удивлении. А-а, ӈй эрэк уркув көӈкэ̄н! — О, кто это стучится в дверь!
   ```

В этой задаче разметка максимально упрощена, нужно использовать только тэг `[b]`, чтобы выделить словарную форму в итоговом определении.

Подзадачи:

- сформировать словарь нужного вида
- итоговый файл сохранить с расширением `dsl`.

Бонус:

- попробовать развести одну словарную статью с римскими цифрами на разные словарные статьи. Например, статья "АБА" должна превратиться в две статьи.

Было в `json`:

```
"АБА": "I обращ. зап. папа, отец (в языке детей); ср. ама.  «Аба! Бӣ-дэ хиннюн көснэ̄м сэ̄м», — гө̄нни көчукэн Мико. -- «Папа, и я хочу пойти с тобой пасти оленей», — говорит маленький Коля. II, АБАЛКАН так говорят дети про что-л. красивое."
```

Стало в `dsl`:

```
АБА
  [b]АБА[/b] обращ. зап. папа, отец (в языке детей); ср. ама.  «Аба! Бӣ-дэ хиннюн көс
нэ̄мсэ̄м», — гө̄нни көчукэн Мико. -- «Папа, и я хочу пойти с тобой пасти оленей», — гово
рит маленький Коля.

АБА
  [b]АБА[/b], АБАЛКАН так говорят дети про что-л. красивое.
```

P.S. Ссылки, не важные для решения задачи:

- DSL
- GoldenDict

3. `dict()`
`with open(…) as …:` `'w'` `json.dump() / json.dumps()`
`input()` `with open(…) as …:` `'r'` `json.load() / json.loads()`
Создайте словарь `transliteration` для транслитерации русских слов в латиницу (для каждой *буквы* описано, на что её транслитерировать) и запишите его в JSON-файл `transliteration.json`. Считывайте файл в память при запуске программы. Напишите

код, который принимает на вход строку, транслитерирует и печатает результат. При создании словаря можно ориентироваться на эту табличку, например, тогда строка `жаба` транслитерируется в `zhaba`.

4. `with open(…) as …:` `.readlines()` `int()` `.writelines()`

   Напишите программу, которая сортирует натуральные числа, записанные в файле `task_5.txt`, по убыванию суммы цифр в десятичной записи числа. Числа, у которых одинаковая сумма цифр, должны располагаться в том же порядке, в котором они были в исходном файле. Количество чисел неизвестно, гарантируется, что оно меньше 10000. Отсортированные числа нужно записать в файл `answer.txt`.

   Входные данные

   Натуральные числа записаны в файле `task_5.txt` в столбик, по одному в строке. Ввод заканчивается тогда, когда заканчиваются данные в файле.

   Выходные данные

   Программа должна вывести в файл `answer.txt` все прочитанные числа, отсортированные по убыванию суммы цифр в десятичной записи числа. Числа, у которых одинаковая сумма цифр, должны располагаться в том же порядке, в котором они были в исходном файле.

5. `wget.download()` `with open(…) as …:` `.readlines()` `.split()` `.strip()` `dict()` `with open(…) as …:` `'w'` `json.dump() / json.dumps()`

   У вас есть тектовый файл с героями саги про хоббитов. Каждая строка состоят из героя, ссылки на него в *fandom.com* и его расы, разделенных запятыми (на самом деле, так выглядят CSV-файлы). Создайте из этого файла словарь и запишите его в JSON-файл вида:

```
{
    "Adanel": {
        "link": "http://lotr.wikia.com//wiki/Adanel",
        "race": "Human"
    },
    "Adrahil I": {
        "link": "http://lotr.wikia.com//wiki/Adrahil_I",
        "race": "Human"
    }
}
```

6. `with open(…) as …:` `'r'`
   `.split()` `.lower()` `.strip(string.punctuation)` `dict()`
   `key=itemgetter(1)` `sorted()`
   `with open(…) as …:` `'w'` `json.dump() / json.dumps()`
   Составьте частотный словарь `freq_dict` слов для текста песни (`слово : кол-во употреблений в тексте`). Перед составлением словаря приведите текст к нижнему регистру и удалите знаки препинания. Выведите 10 наиболее часто встречающихся слов.
   Текст песни любой на ваше усмотрение (или же можно один из этих: `you.txt` , `we.txt`). Храните его в файле и считывайте с файла. Сам полученный словарь запишите в файл `song_dict.json`