работа на уроке (03.03)

1. `def html_to_md()`
   `.replace()`
   Для того, чтобы выделить часть текста *курсивом* или **жирным** шрифтом в MarkDown (`.md`) файлах используются звездочки: одинарные (`*курсив*`) и двойные (`**жирный**`), — или нижние подчеркивания: также одинарные (`_курсив_`) и двойные (`__жирный__`). В HTML-документах (`.html`) для курсива (*italics*) используются парные теги `<i>…</i>`, например `<i>курсив</i>`, а для жирного шрифта (*bold*) — `<b>…</b>`, например `<b>жирный</b>`.
   Напишите функцию `html_to_md()`, которая принимала бы на вход HTML-код, меняла все теги форматирования на звездочки / нижние подчеркивания и позвращала бы файл в формате `.md`.

   HTML-код

   ```
   HTML allows us to add <b>bold</b> to any text that we want to
    <b>emphasize</b>.  We can also add <i>italic</i> text or a
    combination of both <b><i>bold and italic</i></b>.
   ```

   MarkDown-код

   ```
   HTML allows us to add **bold** to any text that we want to
    **emphasize**.  We can also add *italic* text or a
    combination of both ***bold and italic***.
   ```

2. `def ispangram()`
   `set()`
   Напишите функцию `ispangram()`, которая принимает на вход строчку и возвращает `True` или `False` в зависимости от того, является ли эта срока панграммой (короткий текст, использующий все буквы алфавита, по возможности не повторяя их). При этом панграмма может быть как англоязычной, так и русскоязычной!

Примеры панграмм на русском и английском языках:

```
Южно-эфиопский грач увёл мышь за хобот на съезд ящериц.
Sphinx of black quartz, judge my vow.
```

3. `def dice()`
   `random.randint()`

   Напишите симулятор игральных кубиков. У функции `dice()` не должно быть аргументов, однако после ее запуска должно выводиться (просто через `print()`) рандомные значения, “выпавшие” на двух кубиках, и их сумму. Отдельно поздравляйте пользователя, если ему выпал дубль (два одинаковых числа)!

   Примеры запуска:

```
>>> dice()
Let's roll the dice!
Dice One: 1
Dice Two: 6
Total: 7
```

```
>>> dice()
Let's roll the dice!
Dice One: 5
Dice Two: 5
Total: 10
Congrats! It's a double!
```

4. `def is_balanced()`
   `string[]` `for-loop`

   Напишите функцию `is_balanced()`, которая вычислит все ли открытые скобки правильно закрыты.

   Примеры запуска:

```
is_balanced('(()')
False
```

```
is_balanced('))((')
False
```

```
is_balanced('(()())')
True
```

3. Дан `.md` файл в формате

```
# заголовок
text1
## подзаголовок
text2
## подзаголовок
text3

# ещё заголовок
## ещё подзаголовок
### и ещё
text4
```

Пожалуйста, научитесь превращать его (и ему подобных) в `json`, где вложенность заголовков — это вложенность документов. Формат должен стать таким:

```
[
    {
        "title": "заголовок",
        "text": "text1",
        "children": [
            {
                "title": "подзаголовок",
                "text": "text2",
                "children": []
            },
            {
                "title": "подзаголовок",
                "text": "text3",
                "children": []
            }
        ]
    },
    {
        "title": "ещё заголовок",
        "text": "",
        "children": [
```

```
            {
                "title": "ещё подзаголовок",
                "text": "",
                "children": [
                    {
                        "title": "и ещё",
                        "text": "text4",
                        "children": []
                    }
                ]
            }
        ]
    }
]
```

Ваше решение должно содержать функцию `md2json()`. Она должна принимать два аргумента:

1. имя `.md` файла, который надо превратить в `json`
2. имя файла, в который должен быть записан результирующий `json`.

Примечание:

- Игнорируйте пустые строки.
- Вложенность определяется количеством решёток.

4. Дано:

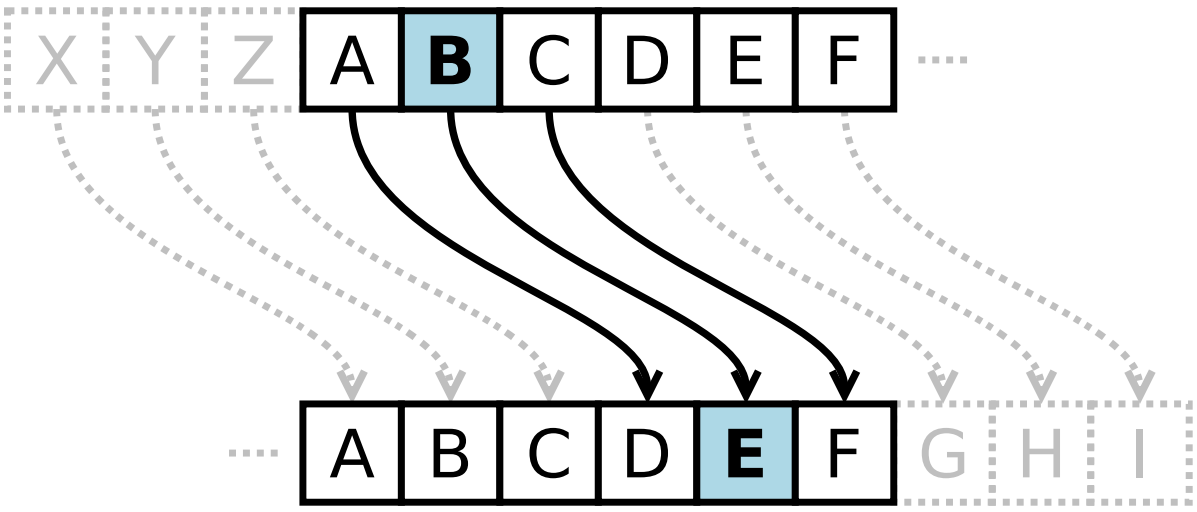
Ваша задача — написать две функции `encode()` и `decode()`, которые позволят обмениваться зашифрованными сообщениями с помощью кнопочного телефона.

Функция `encode()` принимает на вход слово, на выходе выдает последовательность цифр, соответствующие позициям букв в кнопочном телефоне.
Например, `carrot` транформируется в `227768`.

Функция `decode` получается на вход последовательность цифр `23` и переводит ее в последовательность из всех возможных
комбинаций `["ad","ae","af","bd","be","bf","cd","ce","cf"]`. Далее, из всех вариантов выбирается та комбинация(-и) которая(-ые) существуют в языке (для `23` — это `"be"`). Определить это можно по частотности слова в Брауновском корпусе (скачать отсюда).

На выходе функции `decode()` ожидается список кортежей. Первым элементом кортежа является декодированное слово-кандидат, вторым элементом — его частотность в Брауновском корпусе. Слова, не найденные в корпусе в ответ не включать. Если ни одно слова не удается собрать, то возвращать пустой список.

5. `def cypher()`
   `ord()` `chr()` `for-loop`
   Думаю, все знают, что такое Шифр Цезаря: это тот самый, где мы сдвигаем буквы в одну из сторон, например так:



Напишите функцию `cypher`, которая принимала бы на вход строку, а возвращала бы ее в зашифрованном виде со сдвигом 2 (a → c, b → d…).
P. S. После этого можете написать функцию `decypher`, которая помогла бы вам расшифровать эту строчку:

```
"g fmnc wms bgblr rpylqjyrc gr zw fylb. rfyrq ufyr ammnsrcpq ypc dmp. bmgle gr gl zw
 fylb gq glcddgagclr ylb rfyr'q ufw rfgq rcvr gq qm jmle. sqgle qrpgle.myicrpylq() gq
pcammmclbcb))"
```

P. P. S. Напишите функцию `cypher_upgrade`, которая принимала бы на вход не только строку на шифровку, но и число, равное тому, на сколько надо сдвигать все буквы.