

#W4PA: Inverted Index CLI with logging

1. Описание задания	2
2. Инвертированный индекс (Inverted Index)	2
3. Описание данных	3
4. Задания	4

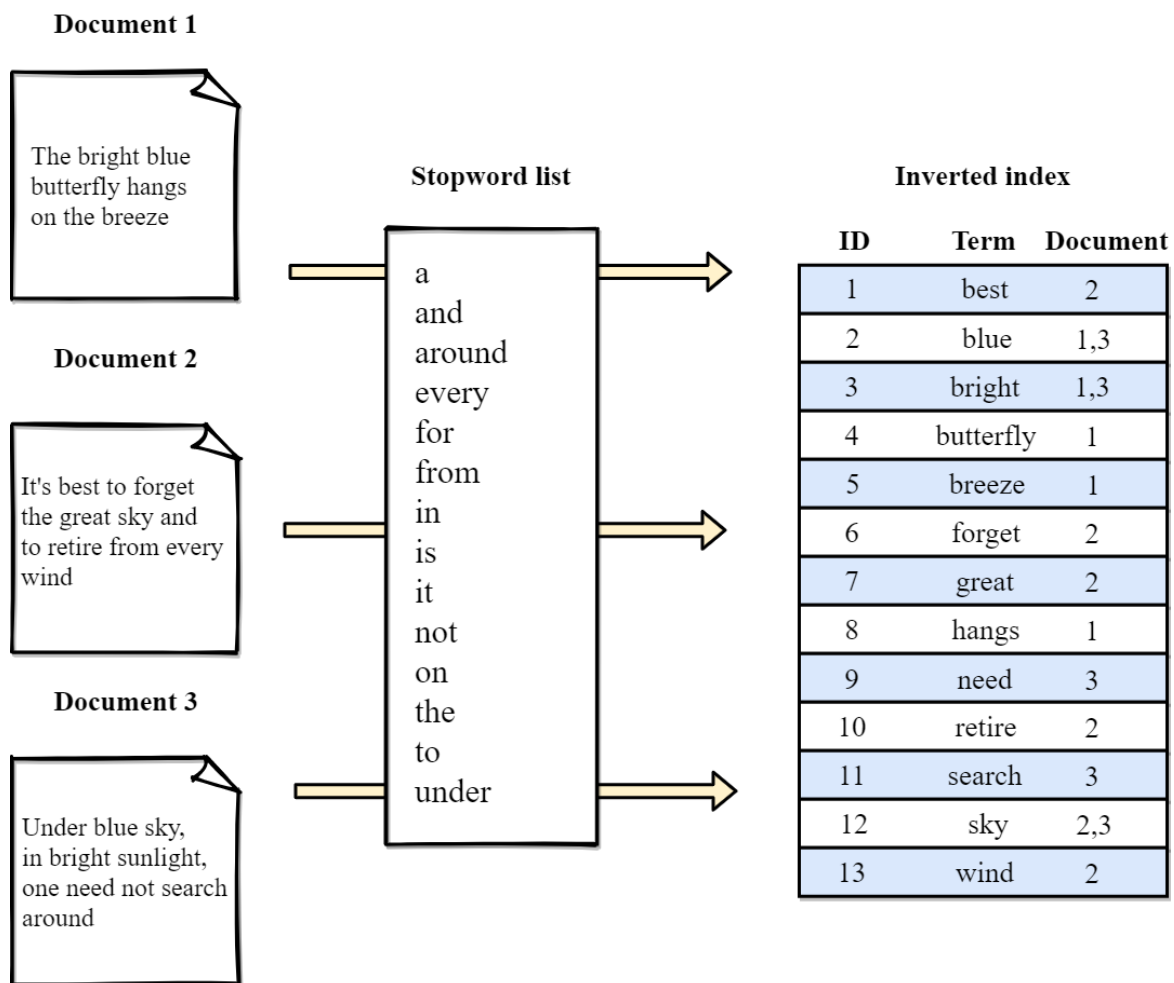
1. Описание задания

В этом задании вам нужно настроить логирование для консольной утилиты по работе с инвертированным индексом. Цель задания:

1. Научить настраивать логирование;
2. Научить его тестировать;

2. Инвертированный индекс (Inverted Index)

Инвертированный индекс представляет собой словарь, где ключами являются слова (термы), а значениями - списки идентификаторов документов, в которых указанный терм встречается (см. Рис. 1).



(Рис. 1) Инвертированный индекс

Такая структура позволяет поисковым системам найти страницы в интернете, которые могут быть релевантны пользовательскому запросу. Вам будет предоставлен датасет из документов и по этому датасету нужно построить инвертированный индекс. Консольное приложение должно предоставлять возможность:

1. Построить инвертированный индекс и сохранить его на диске используя различные стратегии (см. [argparse:add_argument:choices](#) со значением по умолчанию) - `"inverted_index.py build ..."`;
2. Удалить стоп-слова при построении индекса - `"inverted_index.py build --stop-words <path> ..."`;
3. Найти документы, соответствующие поисковым запросам. Если в запросе указаны слова "Python" и "code", то нужно вывести только те документы, которые содержат **оба** этих слова. (см. `action='append'`) - `"inverted_index.py query --json-index <path> --query <word> [<word> ...] --query <word> [<word> ...] ..."`

3. Описание данных

3.1 Дамп Википедии

- Формат: текст
- В каждой строке находятся следующие поля, разделенные знаком табуляции:
 1. INT - id статьи,
 2. STRING - текст статьи,

Пример:

```
12      Anarchism      Anarchism is often defined as a political
philosophy which holds the state to be undesirable, unnecessary, or
harmful.
```

3.2 Стоп-слова

- Формат: одно стоп-слово на строчку

Пример:

```
...
wherein
whereupon
wherever
...
```



4. Задания

Prerequisites:

1. Настройка окружения: <https://github.com/big-data-team/python-course>
2. Датасеты: <https://github.com/big-data-team/python-course#study-datasets>

Возьмите за основу решение задания HW #01: Python CLI Application

С учетом примеров и лайфхаков, показанных на этой неделе:

1. Добавьте логирование в файл и stderr. Настройте логирование с помощью Python объектов `logging.{Formatter,*Handler}` и им подобных. Протестируйте поведение логирования с помощью `pytest:caplog`. Убедитесь, что тесты работают при удалении `pytest.ini` и удалении все опций `--log` из флагов для `pytest`.
2. Научитесь правильно считывать и сохранять `ascii` картинки в YAML файлах. Для вдохновения посмотрите на `ascii` картинки на сайте <https://www.asciart.eu/>.
3. Настройте логирование с помощью YAML конфига.
4. Добавьте возможность выводить логи на экран при указании флага `verbosity`¹:
 - a. `-v` - показывать на экране лог сообщения уровня `WARN+`
 - b. `-vv` - показывать на экране лог сообщения уровня `INFO+`
 - c. `-vvv` (и больше) - показывать на экране лог сообщения уровня `DEBUG+`

Материалы для погружения:

1. <https://docs.python.org/3/howto/logging.html>
2. <https://docs.python.org/3/library/logging.config.html>
3. <https://docs.python.org/3/howto/logging-cookbook.html>

Всем удачи!

¹ См. action "count" для `argparse`