

# HW #02: Stackoverflow Analytics

**Об открытии сдачи сообщим отдельно**

---

1. Описание задания	<b>2</b>
2. Критерии оценивания	<b>2</b>
3. Задача: Stackoverflow Analytics	<b>2</b>
4. Общие рекомендации	<b>5</b>
5. Правила оформления задания	<b>6</b>

---



## 1. Описание задания

В данном ДЗ нужно написать приложение для проведения аналитики постов Stackoverflow. Приложение предоставляет консольный интерфейс для ответа на вопросы о самых популярных темах для обсуждения за указанный период (годы).

## 2. Критерии оценивания

Балл за задачу складывается из:

- **50%** - правильная реализация аналитического инструмента
- **20%** - правильная реализация логирования
- **20%** - качество покрытия тестами, точная формула:
  - $20\% \times \min([\text{test\_coverage} / 0.8], 1.0)$
- **10%** - качество кода (pylint), точная формула:
  - $10\% \times \min([\text{lint\_quality} / 8.0], 1.0)$

Бонусы и штрафы:

- **100%** за плагиат в решениях (всем участникам процесса)
- **5%** за каждую повторную посылку

## 3. Задача: Stackoverflow Analytics

Приложение (`stackoverflow_analytics.py`) должно предоставлять следующий интерфейс<sup>1</sup>:

- чтение датасета в формате xml-lines (кодировка: utf-8)
- чтение стоп-слов (кодировка: koi8-r<sup>2</sup>)
- вывод аналитики на экран
- логирование поведения системы в log-файлы в указанном формате

```
$ python3 stackoverflow_analytics.py \  
  --questions /path/to/dataset/questions.xml \  
  --stop-words /path/to/stop_words_in_koi8r.txt \  
  --log-file /path/to/log_file.log
```

---

<sup>1</sup> Должны поддерживаться как абсолютные, так и относительные пути

<sup>2</sup> В связи с тем, что не все разобрались с кодировками по результатам первого ДЗ



```
--queries /path/to/queries.csv
```

## Входные данные

Stackoverflow:

- пример файла доступен по ссылке для скачивания - [здесь](#);
- формат: xml-lines
- в каждой строке xml следующего вида

```
<row Id="4188365" PostTypeId="1" AcceptedAnswerId="4188400"
CreationDate="2010-11-15T20:09:58.970" Score="1" ViewCount="2325"
Body="<p>I would like to declare a variable 'XMLOutput' and have it
produce the contents of a table in XML format. If you could provide a really
simple example I could work off of I would really appreciate it. I tried using
the <code>xmlelement()</code> but could not get it to
work.</p>&#xA;" OwnerUserId="508703" LastEditorUserId="13302"
LastEditDate="2010-11-15T20:43:01.363"
LastActivityDate="2013-10-18T07:48:48.510" Title="SQL Server 2008: How
to use SQL to output XML from Query?"
Tags="<sql><sql-server><xml><tsql><sql-server-200
8>" AnswerCount="3" CommentCount="1" FavoriteCount="2" />
```

Интересующие поля:

- **PostTypeId** - 1 для вопросов и 2 для ответов
- **CreationDate** - дата создания поста
- **Score** - сумма upvotes и downvotes за пост, вес поста
- **Title** - заголовок поста

Стоп-слова в кодировке "koi8-r":

- пример файла английских стоп-слов доступен - [здесь](#);
- формат: текст
- в каждой строке одно стоп-слово в кодировке "koi8-r"

Запрос на аналитику:

- формат: csv
- в каждой строке:
  - start\_year, end\_year, top\_N



- `start_year` - год для начала проведения аналитики (включительно)
- `end_year` - год для начала проведения аналитики (включительно)
- `top_N` - сколько слов вывести

## Суть анализа

Попробуем понять, какие темы были интересны на Stackoverflow в определенные промежутки времени. Будем исходить из серии “наивных” предположений:

- 1) Количество вопросов по теме обуславливает, в основном, популярность темы.
- 2) Популярность словосочетания темы отражается на популярности отдельных его слов, поэтому не будем заострять внимание на словосочетаниях, а будем угадывать тему по облаку популярных слов.
- 3) Число голосов за вопрос показывает, сколько еще людей считают, что вопрос правильно задан, то есть, насколько популярна тема.
- 4) Суммируя предыдущие предположения: будем считать, что значение популярности слова - это сумма всех оценок вопросов, в заголовке которых встретилось слово 1 или более раз. Математическим языком:

$$word\_score(word) = \sum_{\substack{post \in posts \\ word \in post.Title, post.PostTypeId = 1}} post.Score$$

## Выходной формат

По результатам выполнения приложения, `stdout` должен содержать только ответы на запросы в формате `jsonlines`<sup>3</sup> (всю остальную вспомогательную информацию пишите в `stderr` или в логи). Ответ на запрос - топ-N слов и их `word_score`, встречавшихся в вопросах на Stackoverflow за указанный период (см. пример ниже).

Предобработка `Title` и выделение слов делается следующим кодом:

```
re.findall("\w+", title.lower())
```

Формат `jsonline`:

---

<sup>3</sup> Каждая строка файла - json-документ



```
{“start”: start_year, “end”: end_year, “top”: [[“top_1_word”,  
top_1_word_score), [“top_2_word”, top_2_word_score], ..., [“top_N_word”,  
top_N_word_score]]}
```

Пример:

- Пусть список стоп-слов содержит слова is, than.
- у нас есть три вопроса на Stackoverflow:
  1. title="Is SEO better better better done with repetition?", score="10", year=2019
  2. title="What is SEO?", score="5", year=2019
  3. title="Is Python better than Javascript?", score="20", year=2020
- Нас просят предоставить аналитику за следующие периоды:  
2019,2019,2  
2019,2020,4
- Мы должны вывести на первый запрос 2 наиболее частых слова в заголовках за 2019 год в виде json в следующем формате:  
{“start”: 2019, “end”: 2019, “top”: [[“seo”, 15], [“better”, 10]]}  
Поскольку у нас было несколько кандидатов на top-2 вывели тот, который меньше лексикографически (“better” < “done” < “is” < “repetition”).
- На второй запрос выведем:  
{“start”: 2019, “end”: 2020, “top”: [[“better”, 30], [“javascript”, 20], [“python”, 20], [“seo”, 15]]}

На выходе ожидаются два лог-файла:

1. `stackoverflow_analytics.log` - содержит журнальную информацию со всеми событиями (и следующими уровнями):
  - a. INFO: process XML dataset, ready to serve queries
  - b. DEBUG: got query "start,end,top\_N"
  - c. WARNING: not enough data to answer, found top\_K words out of top\_N for period "start,end"
  - d. INFO: finish processing queries
2. `stackoverflow_analytics.warn` - содержит только предупреждения при обработке аналитических запросов

## 4. Общие рекомендации

При решении задач старайтесь следовать следующим рекомендациям:

- держите уровень покрытия кода тестами на уровне 80+%, следуйте TDD (сначала тесты, потом реализация);
- отделяйте фазу рефакторинга от фазы добавления новой функциональности, т.е.
  - фиксируем функциональность, все тесты зеленые;
  - проводим рефакторинг;
  - по окончании фазы рефакторинга снова все тесты зеленые;
- следите за скоростью выполнения unit-test'ов, несколько секунд – это хорошо, в противном случае нужно уменьшать размер тестируемых датасетов, делать правильный Mock внешних зависимостей или долгих вычислений (e.g. `time.sleep`);
- следите за качеством кода и проверяйте “глупые” ошибки с помощью pylint, следите за поддерживаемостью и читаемостью кода (см. Clean Code и [Google Python Style Guide](#));
- логируйте поведение приложения (см. [logging howto](#) + [logging cookbook](#)).

## 5. Правила оформления задания

- Выполненное ДЗ запакуйте в архив **MADEPY20Q4\_<Surname>\_<Name>\_HW2.zip**, например, для Алексея Драля -- MADEPY20Q4\_Dral\_Alexey\_HW2.zip. Если ваше решение лежит в папке `my_solution_folder`, то для создания архива `hw.zip` на Linux и Mac OS выполните команду:
  - `zip -r hw.zip my_solution_folder/*`
- На Windows 7/8/10: необходимо выделить необходимое для сдачи содержимое директории `my_solution_folder/` нажать правую кнопку мыши на одном из выделенных объектов, выбрать в открывшемся меню "Отправить >", затем "Сжатая ZIP-папка". Теперь можно переименовать архив.



- Перед проверкой убедитесь, что дерево вашего архива выглядит так (в том числе не должно быть лишних файлов и директорий):
  - | MADEPY20Q4\_<Surname>\_<Name>\_HW2.zip
  - | ---- task\_<Surname>\_<Name>\_stackoverflow\_analytics.py
  - | ---- test\_<Surname>\_<Name>\_stackoverflow\_analytics.py<sup>4</sup>
  - При несовпадении дерева вашего архива с представленным деревом ваше решение не будет возможным автоматически проверить, а значит, и оценить его.
- Для того, чтобы сдать задание необходимо:
  - Зарегистрироваться и залогиниться в сервисе [Everest](#)
  - Перейти на страницу приложения "[BigData Team | MADE Python Grader](#)"
  - Выбрать вкладку Submit Job (если отображается иная).
  - Выбрать в качестве "Task" значение: "HW2: Stackoverflow Analytics"<sup>5</sup>
  - Загрузить в качестве "Task solution" архив с решением
  - В качестве Sender ID указать свой индивидуальный id слушателя (если после объявления о том, что всем id разосланы, вы свой не получили, свяжитесь с нами).

Любые вопросы / комментарии / предложения:

- по работе тестирующей системы просьба писать на почту [grader@bigdatateam.org](mailto:grader@bigdatateam.org);
- по заданиям и в целом по курсу - в [Discord-канал](#) курса (#python). [Приглашение на сервер](#), если еще не успели присоединиться.

---

<sup>4</sup> Тесты вашего приложения, которые можно запустить с помощью "PYTHONPATH=. pytest test\_....py".

<sup>5</sup> Сервисный ID: python.stackoverflow\_analytics