BIGDATA TEAM

HW #03: Inverted Index App

1. Описание задания

В этом задании вам нужно написать консольное приложение на Python для работы с инвертированным индексом. Консольное приложение означает, что предоставляется не только консольный интерфейс (HW “Inverted Index CLI”), но еще и тесты на CLI в комплекте (см. pytest:capsys). Сам функционал библиотеки также должен быть расширен и предоставлять возможность:

1. Эффективно сжать индекс с помощью модуля struct;
2. Работать с файлами и потоками в разных кодировках (utf-8 и cp1251).

1.1. Инвертированный индекс (Inverted Index)

Вводные про инвертированный индекс представлены в описании задания “Inverted Index Lib” (учебный модуль про pytest).

1.2. Входные данные

Дамп Википедии

- Формат: текст
- В каждой строке находятся следующие поля, разделенные знаком табуляции:
    1. `INT` - id статьи **в диапазоне 0 .. 65,535**,
    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.
```

1.3. Требования к реализации

Возьмите за основу решение задания “Inverted Index CLI”. На основе решения этого задания, консольный интерфейс библиотеки будет предоставлять возможность:

1. Получить подсказку (help-string) по его использованию;
2. Построить дамп инвертированного индекса;

3. Использовать дамп инвертированного индекса для обработки поисковых запросов.

В дополнение к этому, реализуйте дополнительную функциональность и предоставьте соответствующий консольный интерфейс:

4. Построение эффективного дамп инвертированного индекса с помощью struct;
5. Обработка поисковых запросов в кодировках utf-8, cp1251 в режиме “bulk” (чтение большого числа запросов из файла или потока данных).

Спецификация на консольный интефейс:

1. Получение подсказки (help-string) должно содержать подстроку “Inverted Index CLI”;

```
$ python3 task_*_inverted_index.py -h
$ python3 task_*_inverted_index.py --help
```

2. Должна быть предоставлена возможность использовать разные стратегии сохранения инвертированного индекса на жестком диске (json и struct). По умолчанию должна быть использована эффективная реализация (“struct”).

```
$ python3 task_*_inverted_index.py build --strategy json --dataset
/path/to/dataset --output /path/to/inverted.index
$ python3 task_*_inverted_index.py build --strategy struct --dataset
/path/to/dataset --output /path/to/inverted.index
$ python3 task_*_inverted_index.py build --dataset /path/to/dataset
--output /path/to/inverted.index
```

3. Обработка поисковых запросов в режиме “bulk”. Проверяться будет только struct-часть CLI вашей библиотеки.

```
$ python3 task_*_inverted_index.py query --index
/path/to/inverted.index --strategy struct --query <word> [<word> ...]
--query <word> [<word> ...] ...
$ python3 task_*_inverted_index.py query --index
/path/to/inverted.index --strategy json --query first query [--query
the second query]
```
[1]

[1] Консоль будет работать в кодировке utf-8

```
$ python3 task_*_inverted_index.py query --index
/path/to/inverted.index --query-file-utf8 /path/to/quries.txt
$ cat /path/to/quries.txt | python3 task_*_inverted_index.py query
--index /path/to/inverted.index --strategy struct --query-file-utf8 -
$ python3 task_*_inverted_index.py query --index
/path/to/inverted.index --strategy struct --query-file-cp1251
/path/to/quries.txt
$ cat /path/to/quries.txt | python3 task_*_inverted_index.py query
--index /path/to/inverted.index --strategy struct --query-file-cp1251 -
```

По результатам “обстрела” stdout должен содержать **только** ответы на запросы (всю остальную вспомогательную информацию пишите в stderr или в логи). Ответ на запрос - список идентификаторов документов (статей Википедии), разделенных запятыми. Пример:

- запрос “`--query long query`” состоит из двух слов “`long`” и “`query`”;
- допустим в датасете только 3 документа 151, 13, 3998 содержат **одновременно оба** этих слова, тогда ваш ответ: “151,13,3998”. Порядок предоставленных документов в ответе не важен (может быть любым). Но проверяется, что Вы нашли абсолютно все нужные документы и ничего лишнего;
- если на поисковый запрос не найдено ни одного документа, то нужно выводить пустую строку.

Одновременный вызов `--query-file-cp1251` и `--query-file-utf8` должен приводить к ошибке (return code приложения не должен быть равен 0, см. argparse: mutual exclusive group). Также нельзя одновременно использовать аргументы `--query-file-cp1251` (или `utf8`) и `--query`.

2. Рекомендации

При решении задач, старайтесь следовать следующим рекомендациям:

- следите за качеством кода и проверяйте “глупые” ошибки с помощью pylint, следите за поддерживаемостью и читаемостью кода;
- держите уровень покрытия кода тестами на уровне 80+%, следуйте TDD (сначала тесты, потом реализация);
- отделяйте фазу рефакторинга от фазы добавления новой функциональности.
    - 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 или разделять тесты на фазы (см. обсуждение про mark.slow);

3. Критерии оценивания

Балл за задачу складывается из:

- **40%** - реализация функционала CLI для обработки поисковых запросов в режиме bulk (режим работы из файла в кодировках utf-8, cp1251, режим работы с STDIN и с помощью передачи аргументов через `--query`)
  - запросы типа query должны отрабатывать в течение 5 минут на представленном в описании ДЗ датасете
- **10%** - реализация функционала CLI для построения инвертированного индекса:
  - запросы типа build должны отрабатывать в течение 5 минут на представленном в описании ДЗ датасете
- **20%** - реализация функционала CLI для построения инвертированного индекса и эффективность сжатия индекса с помощью struct, более точная формула:
  - 20% x min(1.5, 12[2] MiB / compression_size)
- **20%** - качество покрытия тестами
  - оценка качества проводится автоматически вызовом pytest:
    - `PYTHONPATH=. pytest -v --cov=task_*_inverted_index test_*_inverted_index.py`
    - уровень покрытия тестами должен быть выше 80%
    - проверяем код Python версии 3.7 с помощью `pytest==6.0.1`
    - точная формула: 20% x min([test_coverage / 0.8], 1.0)
- **10%** - поддерживаемость и читаемость кода
  - в общем случае см. Clean Code и Google Python Style Guide
  - оценка качества будет проводиться автоматическим вызовом pylint:
    - `pylint task_*.py`
    - качество кода должно оцениваться выше 8.0 / 10.0
    - проверяем код Python версии 3.7 с помощью `pylint==2.5.3`
    - точная формула: 10% x min([lint_quality / 8.0], 1.0)

Discounts (скидки и другие акции):

- **100%** за плагиат в решениях (всем участникам процесса)
- **100%** за использование буквосочетаний "exec", "eval", "pylint" (Слова с такими сочетаниями (например, "execute") тоже провоцируют этот штраф). Также в

[2] Приблизительный размер сжатого индекса для датасета Wikipedia (sample), для сравнения политика json даст размер индекса в 26 MiB

этом задании запрещены буквосочетания “pickle”, “zlib”; используйте struct вместо этих библиотек.

- **100%** за посылку решения после hard deadline
- **30%** за посылку решения в после soft deadline и до hard deadline
- **5%** за каждую посылку после 2й посылки в день (каждый день можно делать до 2х посылок без штрафа)

  лучший балл с 1-й попытки: 100%
  лучший балл со 2-й попытки: 100%
  лучший балл с 3-й попытки: 95%
  лучший балл с 4-й попытки: 90%

- до **10%** за эффективность сжатия

4. Инструкция по отправке задания

Оформление задания:

- Код задания (Short name): **HW03:InvertedIndex App**
- Выполненное ДЗ запакуйте в архив **PY-MADE-2021-Q4**_<Surname>_<Name>_HW#**.zip**, пример -- **PY-MADE-2021-Q4**_Dral_Alexey_**HW03**.zip. (Проверяйте отсутствие пробелов и невидимых символов после копирования имени отсюда.[3]) Если ваше решение лежит в папке my_solution_folder, то для создания архива hw.zip на Linux и Mac OS выполните команду[4]:
  - `zip -r hw.zip my_solution_folder/*`
- На Windows 7/8/10: необходимо выделить все 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.
- Решение задания должно содержаться в одной папке.
- Перед проверкой убедитесь, что дерево вашего архива выглядит так:
  - **| PY-MADE-2021-Q4_<Surname>_<Name>_HW03.zip**
  - **| ---- task_<Surname>_<Name>_inverted_index.py**
  - **| ---- test_<Surname>_<Name>_inverted_index.py**
  - **| ---- *.txt**[5]
  - **| ---- conftest.py**

[3] Онлайн инструмент для проверки: https://www.soscisurvey.de/tools/view-chars.php
[4] Флаг -r значит, что будет совершен рекурсивный обход по структуре директории
[5] Архив с тестовыми данными должен занимать **менее 1МБ** пространства на жестком диске

    - 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: MADE Python Grader
    - Выбрать вкладку Submit Job (если отображается иная).
    - Выбрать в качестве “Task” значение: **HW03:InvertedIndex App**[6]
    - Загрузить в качестве “Task solution” файл с решением
    - В качестве Access Token указать тот, который был выслан по почте
- **Перед отправкой задания**, оставьте, пожалуйста, отзыв о домашнем задании по ссылке: https://rebrand.ly/pymade2021q4_feedback_hw. Это позволит нам скорректировать учебную нагрузку по следующим заданиям (в зависимости от того, сколько часов уходит на решение ДЗ), а также ответить на интересующие вопросы.

Внимание: если до дедлайна остается меньше суток, и вы знаете (сами проверили или коллеги сообщили), что сдача решений сломана, обязательно сдайте свое решение, прислав нам ссылку на выполненное задание (Job) на почту с темой письма “Short name. ФИО.”. Например: “**HW03:InvertedIndex App**. Иванов Иван Иванович.” Таким образом, мы сможем увидеть какое решение у вас было до дедлайна и сможем его оценить. Пример ссылки:

- https://everest.distcomp.org/jobs/67893456230000abc0123def

Любые вопросы / комментарии / предложения пишите согласно предложениям на портале.

Всем удачи!

[6] Сервисный ID: python.inverted_index

5. FAQ (часто задаваемые вопросы)

"You are not allowed to run this application", что делать?
Если Вы видите надпись "You are not allowed to run this application" во вкладке Submit Job в Everest, то на данный момент сдача закрыта (нет доступных для сдачи домашних заданий, по техническим причинам или другое). Попробуйте, пожалуйста, еще раз через некоторое время. Если Вы еще ни разу не сдавали, у коллег сдача работает, но Вы видите такое сообщение, сообщите нам об этом.

Grader показывает 0 или < 0, а отчет (Grading report) не помогает решить проблему
Ситуации: • система оценивания показывает оценку (Grade) < 0, а отчет (Grading report) не помогает решить проблему. Пример: в случае неправильно указанного access token система вернет -401 и информацию о том, что его нужно поправить; • система показывает 0 и в отчете (Grading report) не указано, какие тесты не пройдены. Пример: вы отправили невалидный архив (rar вместо zip), не приложили нужные файлы (или наоборот приложили лишние - временные файлы от Mac OS и т.п.), рекомендуется проверить содержимое архива в консоли: `unzip -l your_solution.zip` Если Вы столкнулись с какой-то из них присылайте ссылку на выполненное задание (Job) в чат курса. Пример ссылки: https://everest.distcomp.org/jobs/67893456230000abc0123def

Как правильно настроить окружение, чтобы оно совпадало с тестовым окружением?
1. Если еще не установлено, то установите conda https://docs.conda.io/projects/conda/en/latest/user-guide/install/ 2. Настройте окружение для разработки на основе README.md курса https://github.com/big-data-team/python-course 3. Скачайте необходимые датасеты для выполнения задания https://github.com/big-data-team/python-course#study-datasets

6. Подсказки, если не получается решить ДЗ

Для того, чтобы иметь возможность эффективно пожать (сериализовать) для хранения на жестком диске часто имеет смысл разделить данные по типам. В этом случае, мы сможем использовать эффективные подходы для сжатия (хранения) однотипной информации. Например, вместо того, чтобы хранить в перемешку string и int, мы можем сначала сохранить все string и задать offset (сдвиги в битах от начала файла) для обозначения места в файле, где они будут идти последовательно. А затем иметь блок, где будут храниться все int (что эффективно, поскольку они всегда имеют одинаковый размер и нам не нужна будет дополнительная метаинформация, чтобы указать длину строки для хранения числа типа int).

Частая практика для бинарных форматов - это выделить разделы:

- meta - мета-информация для указания формата и сдвигов для хранения header и body;
- header - информация, где хранится вспомогательная информация для правильного чтения body (например, число идентификаторов (doc_id) на каждое слово)
- body - хранение самих данных.

В связи с вышесказанным предлагается:

- хранить индекс в бинарном, а не текстовом виде (открывать в режиме записи с помощью флага “b”);
- хранить все doc_id последовательно с помощью “unsigned short” (см. диапазон возможных значений doc_id);
- для хранения текстовой информации использовать кодировку utf-8 (можно даже структуру данных сохранить с помощью json.dumps).

Для реализации вам понадобятся следующие методы из модуля struct:

- pack
- unpack
- calcsize

7. Еще подсказки, псевдокод реализации стратегии struct

Псевдокод алгоритма “dump”:

1. Открываем файл на запись в бинарном режиме
2. Итерируемся по парам (слово, doc_ids в которых содержится слово)
3. По каждой паре расширяем массивы
    a. Массив пар (слово, число doc_id в которых встречается слово)
    b. Массив doc_id
4. Создаем header: массив пар перегоняем с помощью json.dumps и кастуем к utf-8
5. Измеряем размер полученного header и пишем этот “unsigned int” первым числом в dump (это будет наша meta)
6. Пакуем и записываем сам header как поток байт рассчитанного размера
7. Пакуем и записываем второй массив как поток “unsigned short”

Псевдокод алгоритма “load”:

1. Открываем файл на чтение в бинарном режиме и считываем его в память
2. Считаем размер (calcsize) “unsigned int”, делаем slice нужно размера из бинарного потока в памяти и делаем распаковку (unpack), чтобы узнать размер header. Обратите внимание на правильное использование unpack:
    a. https://docs.python.org/3/library/struct.html#struct.unpack “The result is a tuple even if it contains exactly one item.” То есть код для чтения должен выглядеть следующим образом:
    ```
    header_size, = struct.unpack(...)
    ```
3. Читаем header, декодируем из utf-8 и распаковываем с помощью json.loads
4. Итерируемся по парам из header, дочитываем нужные doc_id и конструируем ассоциативный массив для инвертированного индекса.