Задача A. Парсинг событийнго лога (rev3)

Имя входного файла: `input.txt`
Имя выходного файла: `output.txt`

Сервис состоит из фронтенда и бекендов. Бекенды бывают нескольких типов. Для увеличения отказоустойчивости бекенды одного типа реплицированы и составляют группу реплик (далее ГР). Бекенды из одной ГР одинаковы, и для формирования выдачи фронтенду достаточно получить ответ хотя бы с одного бекенда из каждой ГР. Фронтенд может сделать несколько попыток обращения к бекендам ГР, пока не получит результат.

У фронтенда есть три фазы обработки запроса:

- Опрос бекендов
- Мерджинг результатов, полученных с бекендов
- Отправка результатов пользователю

В случае, когда фаза опроса бекендов занимает слишком много времени, фронтенд может принудительно завершить опрос и перейти к фазе мерджинга результатов с неполным набором данных.

В процессе работы фронтенд пишет в лог файл события, возникающие при обработке запроса.

Вам нужно по данному логу событий посчитать:

1. 95-й перцентиль времени обработки запросов фронтендом
2. Найдите 10 запросов, в которых фаза отправки результатов пользователю была максимальной. В качестве ответа выдайте 10 идентификаторов запросов.
3. Для каждого бекенда, отметившегося в логе, посчитайте количество обращений к нему, количество и типы ошибок, возникших при работе с ним.
4. Посчитайте количество запросов к фронтенду на которые фронтенд не смог собрать данные со всех ГР.

Формат входных данных

Входной файл состоит из строчек в формате:
`<время события> <идентификатор запроса на фронтенд> <тип события> [<дополнительные параметры>]`
Разделителем полей выступает символ табуляции.

Строки отсортированы по времени события — UNIX timestamp в микросекундах.

Идентификатор запроса это уникальное целое число. Отфильтровав лог по записям с выбранным идентификатором запроса можно узнать всё об обработке соответствующего пользовательского запроса.

Типы событий:

StartRequest
Начало обработки запроса

BackendConnect
Установка tcp соединения с бекендом.
Дополнительные параметры: `<номер ГР> <URL запроса на бекенд>`

BackendRequest
Отправка запроса на бекенд.
Дополнительные параметры: `<номер ГР>`

BackendOk
Отметка успешного получения ответа с бекенда. Работа с ГР завершается.
Дополнительные параметры: `<номер ГР>`

BackendError
Ошибка работы с бекендом. Следует после **BackendConnect** или **BackendRequest**.
Дополнительные параметры: `<номер ГР> <строка с текстом возникшей ошибки>`

StartMerge
Означает конец фазы опроса бекендов и начало фазы мерджа результатов.

StartSendResult
Означает конец фазы мерджа и начало отправки результата пользователю.

FinishRequest
Конец обработки запроса.

Формат выходных данных

Выведите в свободной фоме ответ на поставленные вопросы.

Время обработки пользовательского запроса вычисляется как время между событиями **StartRequest** и **FinishRequest**. Подсмотреть, что такое 95-й перцентиль можно тут: `ru.wikipedia.org/wiki/Квантиль`

Время ответа пользователю можно вычислить как разницу времени между событиями **StartSendResult** и **FinishRequest**.

Бекенды, с которыми производилась работа можно узнать разобрав URL в событии **BackendConnect**. В качестве типа ошибки можно использовать строковое представление ошибки в событии **BackendError**.

Запросы, на которые ответили не все ГР не имеют событий **BackendOK** хотя бы для одной из своих ГР.

Система оценки

- Мы ожидаем, что решение будет оформленно на Python 2.7.
- Можно пользоваться любыми библиотеками с pypi, но не забудьте описать зависимости в `pip-requirements.txt`. Также, мы будем признательны, если, при прочих равных, вы выберете pure python библиотеки себе в зависимости — нам это сильно упрощает проверку.
- В общем случае событийный лог может быть достаточно большим. Программа не должна зачитывать весь лог в память.
- Постарайтесь оптимизировать свою программу по памяти и процессору.
- Программа будет запускаться только на предоставленных вам вариантах входных файлов. Т.е. не нужно как-то валидировать входные данные, рассчитывать на рваные логи и проч.

Пример

input.txt

```
1390950160808136 0 StartRequest
1390950160810164 0 BackendConnect 0 http://backend0-001.yandex.ru:1963/search?
1390950160810179 0 BackendConnect 1 http://backend1-001.yandex.ru:1085/search?
1390950160841530 0 BackendRequest 0
1390950160938308 0 BackendRequest 1
1390950160948308 0 BackendOk 1
1390950161841530 0 BackendError 0 Request Timeout
1390950161842604 0 BackendConnect 0 http://backend0-002.yandex.ru:1126/search?
1390950161928218 0 BackendRequest 0
1390950162464394 0 BackendOk 0
1390950162475798 0 StartMerge
1390950162536865 0 StartSendResult
1390950162890134 0 FinishRequest
```

output.txt

```
95-й перцентиль времени работы: 2081998

Идентификаторы запросов с самой долгой фазой отправки результатов пользователю:
    0

Запросов с неполным набором ответивших ГР: 0

Обращения и ошибки по бекендам:
ГР 0:
    backend0-001.yandex.ru:1963
        Обращения: 1
        Ошибки:
            Request Timeout: 1
    backend0-002.yandex.ru:1126
        Обращения: 1
ГР 1:
    backend1-001.yandex.ru:1085
        Обращения: 1
```

Пояснение к примеру

В этом примере у фронтенда есть две ГР. Работа с ГР 1 завершилась с первой попытки удачно. При работе с нулевой ГР возникла проблема при обращении к бекенду `backend0-001.yandex.ru:1963` и был произведён удачный перезапрос на бекенд `backend0-002.yandex.ru:1126`.

Пример

input.txt

```
1390672572320679 0 StartRequest
1390672572331061 0 BackendConnect 1 http://backend1-001.yandex.ru:1495/search?
1390672572331534 0 BackendConnect 0 http://backend0-001.yandex.ru:1848/search?
1390672572331939 0 BackendConnect 2 http://backend2-001.yandex.ru:1789/search?
1390672572332019 0 BackendConnect 4 http://backend4-002.yandex.ru:1445/search?
1390672572332896 0 BackendConnect 3 http://backend3-002.yandex.ru:1433/search?
1390672572370445 0 BackendRequest 2
1390672572371253 0 BackendRequest 4
1390672572376201 0 BackendRequest 0
1390672572381722 0 BackendRequest 1
1390672572389323 0 BackendRequest 3
1390672572823197 0 BackendOk 2
1390672572832417 0 BackendOk 0
1390672572871253 0 BackendError 4 ParseFromStream error
1390672572881722 0 BackendError 1 Input/output error
1390672572882606 0 BackendOk 3
1390672572882729 0 BackendConnect 4 http://backend4-001.yandex.ru:1002/search?
1390672572891737 0 BackendConnect 1 http://backend1-001.yandex.ru:1495/search?
1390672572901758 0 BackendRequest 1
1390672572915564 0 BackendRequest 4
1390672573269124 0 BackendOk 1
1390672573415564 0 BackendError 4 Connection reset by peer
1390672573425081 0 StartMerge
1390672573483245 0 StartSendResult
1390672574005831 0 FinishRequest
```

output.txt

```

```

Пояснение к примеру

В этом примере фронтенд не дождался ответа 4-ой ГР, т.к. решил, что лучше показать неполный результат, нежели ещё больше заставлять пользователя ждать.