###### Выбор компилятора для языка Java

По умолчанию рекомендуем использовать компиляторы Java 11 и Java 15 для отправки решений. Гарантируется решение на компиляторе "Java 8 + network + json-simple" ТОЛЬКО для задачи А. Для остальных задач решение на данном компиляторе не гарантируется в отличие от Java 11 и Java 15.

A. Состав заказа

|  |  |
| --- | --- |
| Ограничение времени | 2 секунды |
| Ограничение памяти | 512.0 Мб |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Одной из важных частей онлайн-магазинов является обработка заказов. Заказ содержит в себе несколько товаров, каждый из которых приобретается в определенном количестве.

Но состав заказа не является строго фиксированным с момента оформления до момента фактической отправки заказчику. Пользователь может передумать покупать какой-либо товар (весь или частично) или, наоборот, решить увеличить количество какого-либо товара в заказе перед отправкой. В итоге может получиться даже так, что заказ в итоге нет смысла отправлять, так как в нём не осталось товаров.

Всё, что происходит с заказами в системе, фиксируется в формате событий "после изменения **event\_id** в заказе **order\_id** итоговое заказанное количество товара **item\_id** равно **count**, а итоговое отмененное количество равно **return\_count**". В итоге реальное количество товаров для отправки может быть вычислено как разница между **count** и **return\_count**. Разница может получиться отрицательной - в рамках данной задачи это равносильно разнице, равной нулю (в причинах расхождения количеств в событии будет заниматься другой отдел).

Также у товара в заказе есть статус - OK*OK* или CANCEL*CANCEL*. Это независимая от количества товара информация, обозначающая возможность отправки товара в данном заказе - если товар находится в статусе CANCEL*CANCEL*, то его не надо отправлять, сколько бы штук товара не требовалось. Важно отметить, что статус CANCEL*CANCEL* может сменить на OK*OK* в дальнейшем - к примеру, на складе появилась новая партия товара.

Вам требуется обработать список событий и вывести для каждого заказа итоговый список товаров для отправки. Для каждого товара необходимо учитывать **только последнее событие** (событие с максимальным значением **event\_id**) среди событий, содержащих информацию о данном товаре. Если итоговый статус товара - CANCEL*CANCEL*, то он не должен попасть в заказ. В заказ должны попасть только товары с ненулевым итоговым количеством для отправки. Выводить необходимо только заказы, в которых содержится хотя бы один товар для отправки.

Формат ввода

Входные данные представляют собой список событий в формате **JSON**.

Гарантии по формату JSON:

* нет запятых после последнего элемента массива;
* все имена полей и строки обернуты в двойные кавычки.

Гарантий по отступам и переводам строк **нет** - в примерах отступы и переводы строк используются для наглядности.

Обозначим количество событий в списке через N*N*. Гарантируется, что 0≤N≤10000≤*N*≤1000.

Каждое событие содержит следующую информацию (порядок полей не является фиксированным):

* **event\_id**: идентификатор события.
* **order\_id**: идентификатор заказа.
* **item\_id**: идентификатор товара в заказе.
* **count**: Итоговое количество единиц товара в заказе, запрошенное к отправке.
* **return\_count**: Итоговое количество единиц товара в заказе, отмененное к отправке.
* **status**: статус товара в заказе.

Гарантируется, что идентификаторы и количества являются целыми числами (0≤(0≤ **event\_id**, **order\_id**, **item\_id** ≤231−1≤231−1; 0≤0≤ **count**, **return\_count** ≤109)≤109).

Гарантируется, что статус представлен только строками OK*OK* и CANCEL*CANCEL*.

Гарантируется, что идентификаторы событий **уникальны**, но не гарантируется упорядоченность списка событий по возрастанию идентификаторов.

Формат вывода

Выведите в формате **JSON** список заказов для отправки.

Выводить JSON допустимо без дополнительных отступов и переводов строк - в примерах отступы и переводы строк используются для наглядности.

Имена полей необходимо выводить в **двойных кавычках**.

Допустимо выводить запятую после последнего поля объекта или последнего элемента массива.

Каждый заказ должен содержать следующую информацию:

* **id**: идентификатор заказа.
* **items**: список товаров в заказе.

Каждый товар в заказе должен содержать следующую информацию:

* **id**: идентификатор товара.
* **count**: количество товара для отправки.

Заказы и товары в рамках одного заказа можно выводить в любом порядке. Поля одной JSON-сущности так же можно выводить в любом порядке.

Ответ **не должен содержать**:

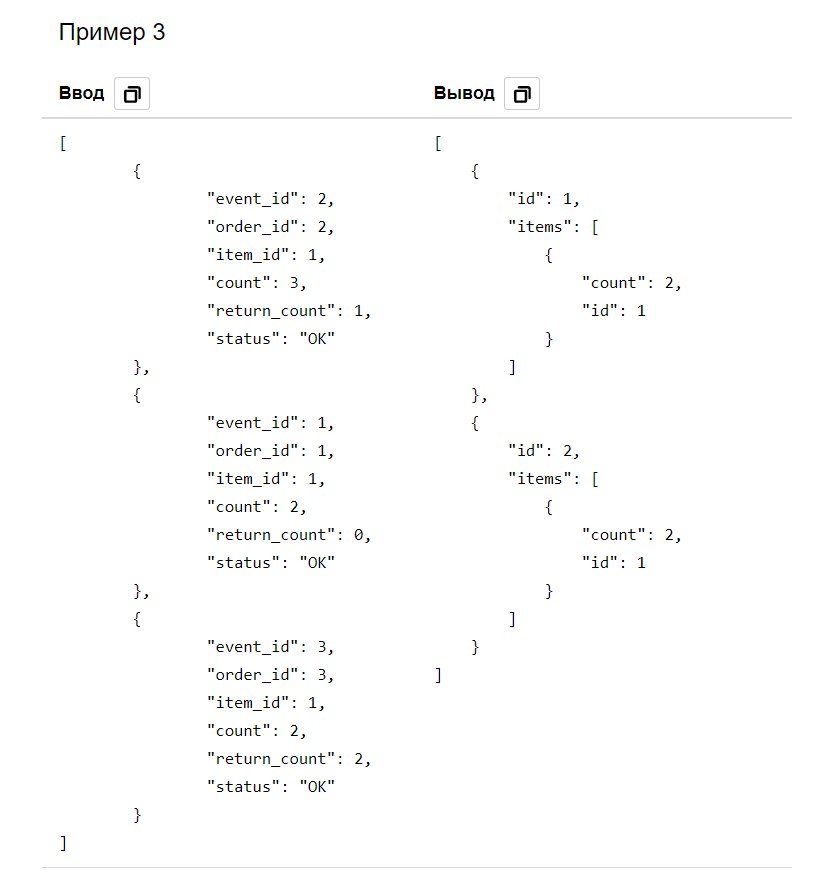
* товары с нулевым количеством для отправки.
* товары с итоговым статусом CANCEL*CANCEL*.
* заказы с пустым списком товаров.

### **Пример 1**

|  |  |
| --- | --- |
| [  {  "event\_id": 1,  "order\_id": 1,  "item\_id": 1,  "count": 1,  "return\_count": 0,  "status": "OK"  },  {  "event\_id": 2,  "order\_id": 1,  "item\_id": 1,  "count": 1,  "return\_count": 0,  "status": "CANCEL"  }  ] |  |







Использовать json-simple 1.1.1